ENHANCED MOVIE RECOMMENDER SYSTEM USING A STATISTICAL APPROACH

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
ÇANKAYA UNIVERSITY

BY

HÜSEYİN BURHAN ÖZKAN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
COMPUTER ENGINEERING

DECEMBER 2010

Title of Thesis: **ENHANCED MOVIE RECOMMENDER SYSTEM USING A STATISTICAL APPROACH**

Submitted by **HÜSEYİN BURHAN ÖZKAN**

Approval of the Graduate School of Natural and Applied Sciences, Çankaya University

[illegible]

Prof. Dr. Taner ALTUNOK
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

M. R. Tolun

Prof. Dr. Mehmet R. TOLUN
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

M. R. Tolun

Prof. Dr. Mehmet R. TOLUN
Supervisor

Examination Date : 01.12.2010

Examining Committee Members

Asst. Prof. Dr. Murat SARAN	(Çankaya Univ.)	[illegible]
Prof. Dr. Mehmet R. TOLUN	(Çankaya Univ.)	M. R. Tolun
Dr. Ayşenur Akyüz BİRTÜRK	(METU)	[illegible]

STATEMENT OF NON PLAGIARISM

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name	: Hüseyin Burhan ÖZKAN
Signature	: [illegible]
Date	: 01.12.2010

ABSTRACT

ENHANCED MOVIE RECOMMENDER SYSTEM USING

A STATISTICAL APPROACH

ÖZKAN, Hüseyin Burhan

M.Sc., Department Computer Engineering

Supervisor : Prof. Dr. Mehmet R. TOLUN

November 2010, 53 pages

With the Web 2.0, which can also be named as the Social Web, securing its position in our lives and spreading the practice of people sharing and collaboratively generating Internet content, as well as presenting new opportunities there emerges a new and complex structure and massive amounts of information that must be brought into the use of people. A part of this information consists of people and communities sharing their tastes on specific entities either implicitly or explicitly. This information consisting of choice of people is not always usable at its raw state and presented to the utilization of people who are a part of the social network by means of systems called recommender systems which employ data mining methods. Recently there is lots of research done in this area but there are still aspects to be studied. In this thesis, the EM algorithm that has been widely used in scientific researches but has not been substantially used for recommender systems is integrated with other collaborative and content based approaches to build an efficient and scalable system. The system is tested using different data sets and it is found that its performance is sufficient in terms of both accuracy and computation time.

Keywords: EM Algorithm, Clustering, Collaborative Recommender System, Content-based Recommender System

ÖZ

İSTATİSTİKSEL BİR YAKLAŞIM KULLANARAK GELİŞTİRİLMİŞ FİLM TAVSİYE SİSTEMİ

ÖZKAN, Hüseyin Burhan

Yükseklisans, Bilgisayar Mühendisliği Ana Bilim Dalı

Tez Yöneticisi : Prof. Dr. Mehmet R. TOLUN

Kasım 2010, 53 sayfa

Sosyal Web olarak adlandırabileceğimiz Web 2.0'ın hayatımızdaki yerini sağlamlaştırması ve insanların paylaşarak ve ortaklaşa bir şekilde internet içeriğini oluşturması uygulamasının yaygınlaşması bize yeni olanaklar sunmakla birlikte ortaya yeni ve karmaşık bir yapı ve insanların kullanımına sunulması gereken büyük miktarda bilgi çıkmaktadır. İşlenmesi gereken bu bilgilerin bir kısmı insanların ve toplulukların doğrudan veya dolaylı yollarla belirli varlıklar hakkındaki beğenilerini paylaşmasından meydana gelmektedir. İnsanların beğenilerinden ibaret olan bu bilgiler her zaman ham haliyle kullanılabilir olmamakla birlikte tavsiye sistemleri adı verilen ve veri madenciliğini kullanan sistemler tarafından işlenerek sosyal ağın parçası olan diğer kullanıcıların kullanımına sunulmaktadır. Son yıllarda bu alanda pek çok çalışma yapılmıştır ancak halen çalışılması gereken yönler vardır. Bu çalışmada, pek çok bilimsel çalışmada kullanılan ancak tavsiye sistemlerinde daha önce fazlaca kullanılmamış olan EM algoritmasını diğer içerik tabanlı ve işbirliğine dayalı yaklaşımlarla birleştirerek etkin ve ölçeklenebilir bir sistem oluşturmaya çalışılmıştır. Sistem farklı veri grupları için test edilerek performansının hem doğruluk hem de işlem süresi açısından yeterli olduğu tespit edilmiştir.

Anahtar Kelimeler: EM Algoritması, Kümeleme, İşbirliğine Dayalı Tavsiye Sistemi, İçerik Tabanlı Tavsiye Sistemi

ACKNOWLEDGEMENTS

I would like to express my gratitude to my supervisor Prof. Dr. Mehmet R. TOLUN for his guidance, support and patience throughout this study. Also I would like to thank to Asst. Prof. Dr. Murat SARAN and Dr. Ayşenur Akyüz BİRTÜRK who helped to make this thesis better.

Also many thanks to my family and friends due to their support and understanding over the course of this challenge.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ARFF	Attribute-Relation File Format
DSAM	Decision Support Accuracy Metrics
EM	Expectation Maximization
IDF	Inverse Document Frequency
ISF	Inverse Sentence Frequency
KDD	Knowledge Discovery and Data Mining
LSI	Latent Semantic Indexing
MAE	Mean Absolute Error
RMSE	Root Mean Square Error
SVD	Singular Value Decomposition
TF	Term Frequency
XML	Extensible Markup Language

CHAPTER 1

INTRODUCTION

In general terms, in the context of data mining, filtering means to process a set of data to acquire or attain useful information according to some predefined criteria. Filtering includes eliminating data that seems redundant. This process is vital in this information age because the amount of data is massive and it is often impossible for the users to cope with this information overload and achieve the data that is necessary without the help of data mining systems and artificial intelligence. Recommender systems are the application of filtering systems and they are used to recommend or help the user find an item among a collection of items that will most probably be liked by the user.

After the introduction of the term of collaborative filtering in the mid 90's [6] and with the efforts of the team that worked on the GroupLens Research Project [1][2][52] the recommender systems have become a subject of general interest and research. Since then there appear lots of research, conferences, thesis studies and competitions done on the subject. The attention to recommender systems however is not purely scientific. Lots of sites and systems emerged that use recommender systems for item recommendation. These employ different recommendation techniques to recommend movies, music, books, articles or work as a shopping assistant.

Although lots of research is done to achieve more accurate, scalable and useful systems there is more room to study. There are various problems related to different types of filtering techniques used and also there are prediction and performance concerns. For example 1 million $ prize Netflix challenge [33] is one good example for the studies to achieve more accurate predictions. Also year 2007 challenge for the traditional KDD cup [20] focuses on two tasks using the Netflix database first to determine which movies will be rated by which users and the number of additional ratings a movie will get in a year.

The subject of this thesis is in general recommender systems and in particular statistical enhancement of movie recommender systems. The second chapter is a brief introduction to the recommender system techniques used. Also details on the problems faced during the process and about the special case of movie recommendation and its real world application are given. Third chapter is on Enhanced Movie Recommender system which is the main topic of this study. In the fourth chapter the system is evaluated and is shown that it has a reasonable performance compared to the other implementations. The last chapter is a brief summary of the study and there will be directions on future work.

CHAPTER 2

RECOMMENDER SYSTEMS

In the first chapter a brief introduction is made to recommender systems here the subject is explained in detail. To build on the previous chapter, from now on when we say recommender system we will be referring to recommending an item to a user by means of using item and user information and user ratings previously given to other items. First section of this chapter is on different methods employed during recommendation, second part is on the problems faced and the third part is on the special case of movie recommendation.

2.1. Methods

There are different methods used while implementing a recommendation system. Three main methods used are content-based (item) filtering, collaborative filtering and hybrid recommendation. In the following subsections we will talk on these methods.

2.1.1. Collaborative (Social) Filtering

Collaborative filtering is the earliest method that is used in item recommendation. This is because the idea behind it is simple; we will like an item if someone with a similar taste as us likes it. This is also the case in everyday life. We ask our friends, who we know that have a similar taste with us, for example about movies before going to a movie. That is a method which is used to maximize the likelihood of going to a movie that will be fancied before even seeing it. A pure collaborative filtering approach doesn’t account for item properties and only employs previous user ratings on the items to give a rating prediction or predict if the user will like the item or not. Collaborative filtering is a very broad area so we will only try to give a brief description of the widely used methods.

Collaborative filtering algorithms have been classified according to different approaches. According to the earlier approach collaborative filtering algorithms are classified in two groups [13]:

i. Memory-based or probabilistic methods which operate over the entire database to make a recommendation
ii. Model based methods which creates a model using the database and then makes a recommendation

Other than that, Schafer et al. suggests to classify collaborative filtering algorithms in two groups of [15]:

i. Probabilistic algorithms which represent probability distributions when computing predicted ratings or ranked recommendation lists
ii. Non-probabilistic algorithms which doesn’t rely on an underlying probabilistic model

It is also possible to categorize collaborative filtering algorithms as:

i. User Based algorithms which try to find similar users that has similar ratings to the target user

ii. Item Based algorithms which try to predict the probable rating by inspecting similar items that the user has rated.

2.1.1.1. Nearest Neighbor Method

This is the simplest and most widely used method used for collaborative filtering. It depends on the idea of finding users who has a likely rating pattern as the target user or finding likely items rated by the user as the target item and makes a prediction using a weighted average of these users' votes or items' ratings. While finding likely neighbors different formulations of correlation, distance and similarity are used.

i. Correlation

In statistical terms correlation is a value between +1 and -1 which shows the degree of association between two variables or data sets. A value of +1 indicates a positive linear relation (association) between the variables and a value of -1 indicates a negative linear association. The absolute value of the correlation being close to 1 indicates a strong relationship between users.

Pearson correlation is one of the most widely used methods in correlation computation. It has the formula in 1.1:

$$\frac{\sum_{i=1}^{n}(X_i-\bar{x})(Y_i-\bar{y})}{ns_x s_y} \qquad (2.1)$$

In the formula $\bar{x}$ and $\bar{y}$ are the means and s_x and s_y are the standard deviations of the two sets. After the correlation is calculated the most useful users are the ones that have high correlation. These users are the ones who have similar taste as the target user and their previous votes will be used while making a prediction about our user.

ii. Distance

While searching for similarity between users a useful approach will be finding distance between users according to some criteria and distance formula. For example, the degree of difference between the ratings given to the same item by two distinct users can be a good criterion. While calculating the distance, different distance formulas with various complexities can be used. As an example Euclidean Distance is the most basic distance formula used for calculating the distance between two vectors or sets;

$$d = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \tag{2.2}$$

The most similar users to the target user are the ones with smaller distances.

iii. Similarity

Similarity computation between two sets of data can be done by treating these two sets as vectors and using the formula for the cosine angle between two vectors [16]. Cosine similarity formula between two vectors $\vec{x}$ and $\vec{y}$ is:

$$\cos(\vec{x}, \vec{y}) = \frac{\vec{x} \cdot \vec{y}}{\|x\| \|y\|} \tag{2.3}$$

$\vec{x} \cdot \vec{y}$ is the vector dot product and $\|x\|$ and $\|y\|$ are vector magnitudes. This formula gives the cosine value between two vectors which ranges between +1 and -1. The value being closer to 1 means the similarity of the two users or items is great.

iv. Prediction

After acquiring the neighbors to the user or the item the second phase is to give predictions on probable user ratings. There are several approaches of generating predictions using neighbors. The two prediction computation methods according to Sarwar et al. are weighted sum and regression [16].

a. Using the weighted average method, prediction is found by taking the average of the ratings of similar users/items multiplying each user/item rating with the corresponding similarity. The formula [16] can be seen in 1.4.

$$\sum_{i=1}^{n} \frac{R_i * W_i}{n} \tag{2.4}$$

Here are R_i is the item rating and W_i is the corresponding weight.

b. The second method of prediction depends on the regression model. This method is used because although two user/item vectors are similar the distance between them can be large [16]. So we use a new variable $\acute{R}_i$ which is obtained by using regression [16].

$$\acute{R}_i = \alpha R_i + \beta + \varepsilon \tag{2.5}$$

Here α and β are the regression parameters computed using the rating data and ε is the regression error.

2.1.1.2. Naïve Bayesian Classifier Model

Naïve Bayesian classifier is a statistical model used to determine the class label C of an item i. Say an item has attribute vector $\{A_i \ , \dots , A_n\}$ and each item belongs to a class $\{C_i, \dots , C_n\}$. The classifier makes a statistical analysis of the data set and then finds probabilities for each attribute belonging to a specific class C[18]. Then for a specific item while finding the class we calculate the class which the item most probably belong to using the attribute probability values. Here we are making the assumption that each attribute probability is independent from each other. Using this assumption, in 1.7 you can see the probability of the item i with attributes A belonging to the class C_i.

$$P(C_i|A) = \frac{P(A|C_i)P(C_i)}{P(A)} \tag{2.6}$$

After we calculate all the class membership probabilities we label the item with the class which has the most probability. Now we can talk about how this method is used for recommendation.

According to the study made by Miyahara et al. [19] each user has a class label of $\{like, dislike\}$ for a specific item and a matrix is formed with other users' ratings as feature values. Then the probability of our current user belonging to each class is calculated using the matrix. On the other hand Breese et al. [13] pursued a different approach. They tried to find if an item will be 'watched' or 'not watched' by a user through looking to the other items user has 'watched' or 'not watched'. While doing this they used a Bayesian Network Approach. An example for this case may be finding “the probability of a user who has watched Star Trek to watch Transformers”.

2.1.1.3. Latent Factor Method and Matrix Factorization

Another popular method employed recently for collaborative filtering is latent factor method with matrix factorization. Latent factor method tries to explain the ratings by characterizing both items and users on, say, 20 to 100 factors inferred from the ratings patterns [21]. For example in the movie domain these factors can range from the factor of comedy in the movie, the amount of cgi used, the movie being realistic or not to complex factors that may not be obvious to humans at first glance.

While finding latent factors matrix factorization over user item rating matrix is a generally employed method. Matrix factorization is a technique used to decompose the Rating matrix R in to product of smaller matrices U and I that is;

$$R \approx U.I \tag{2.7}$$

There are many approaches that try to address this type of recommendation. One example is Latent Semantic Indexing (LSI) with Singular Value Decomposition (SVD) which is originally an information retrieval algorithm. SVD is a dimensionality reduction algorithm which we use to produce low rank approximations [22]. Applying SVD on ratings matrix R;

$$SVD[R] = U \times S \times I^T \tag{2.8}$$

Here R is $\mathrm{m} \times \mathrm{n}$, U is $\mathrm{m} \times \mathrm{m}$, S is $\mathrm{m} \times \mathrm{n}$ I is $\mathrm{n} \times \mathrm{n}$ dimensionality matrices. The rank of R is r. S can be reduced to diagonal matrix S_k of size $\mathrm{r} \times \mathrm{r}$ with all singular values of matrix R as its diagonal entries with 0 rows omitted. Similarly U can be reduced to U_k and I can be reduced to I_k deleting the corresponding rows [24].

Using this reduced S matrix the prediction can be obtained using the following calculations [22];

i. Calculate $\sqrt{[S_k]} = [S_k]^{1/2}$
ii. Compute the matrices $[U_k][S_k]^{1/2}$ and $[S_k]^{1/2}[I_k]^T$
iii. To obtain the prediction for user u and item i calculate the dot product of uth row of $[U_k][S_k]^{1/2}$ and ith column of $[S_k]^{1/2}[I_k]^T$.

LSI technique using SVD is successful especially for large and sparse (see section 2.2) databases.

2.1.2. Content-Based (Item) Filtering

Content or item based recommendation is a method that includes item descriptions and user preferences for these items. According to Pazzani et al. [5], which is a great introduction to content-based recommenders, content-based systems share in common means;

i. for describing the items that may be recommended
ii. for creating a profile of the user that describes the types of items the user likes
iii. of comparing items to the user profile to determine what to recommend.

This method gained interest when the drawbacks of the previous collaborative filtering, which we will talk about in the next part, have appeared. In the Figure 1.1 you can see a sample content base filtering system similar to the one depicted in [7].

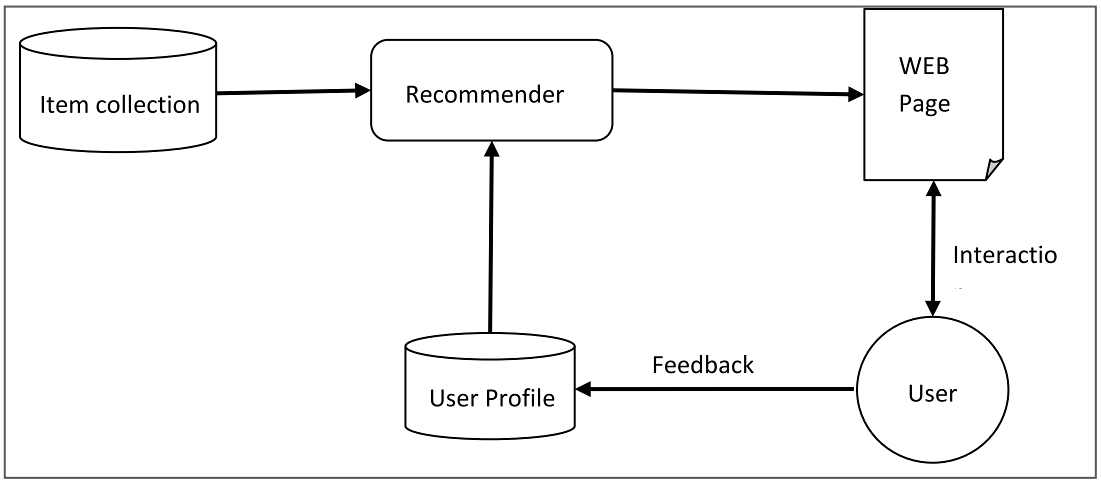


Figure 2.1. Content Base Filtering System

An item that may be recommended is stored in the item collection database. The recommender system compares the items to the items that are on the user profile. The comparison here is made according to some certain criteria such as similarity, novelty, proximity and relevancy [7]. The feedback that is made by the user to the user profile helps us to give better recommendations in time. The important point here is the item representation. As mentioned previously there must be a way for the recommender to understand the item representation and compare the items in the item collection with the items on the user profile. Let’s give more details on the properties of content based filtering.

2.1.2.1. Item Representation

According to Pazzani et al. [5] attribute values of items in the item collection can be structured data where each attribute takes a value among a defined set of data or unrestricted text value which must be processed to *make sense* out of it. Structured data for example can be the genre attribute of a movie, or the writer of a book. This type of data is easier to be valued and compared.

Unrestricted text value can be a comment on a film or plot summary of a film. A human can understand these type of data easily but for a recommender system it is a hard job to extract useful, i.e. comparable and gradable, information out of free text and form a model. So generally text processing and information retrieval techniques are used in this case. The aim here is generally to make a summary of the text and cluster similar items. One of the most employed techniques used for this task is term frequency – inverse sentence frequency (TF-ISF) which is first described in [8]. TF is the frequency, count, of a word appearing in a document. IDF can be calculated by;

$$IDF = log(N/N(w)) \tag{2.9}$$

Here N is the number of documents in the collection and $N(w)$ is the count of documents containing the word. We can see that as $N(w)$ increases IDF decreases.

This method depends on two simple assumptions:

i. A word appearing many times in a document is representative of the document

ii. A word appearing many times in multiple documents has low discriminating power between documents [9].

So we are trying to find words that represent the document well. So for a word to be representative of the document both TF and IDF must be high. So this can be represented as $TF * IDF$ value being high.

2.1.2.2. User Preferences

The backbone of the item based collaboration is determining the user preferences regarding the items and modeling the “taste” of the user. The user profile is constituted for this task. The preference of the user can be deducted from the items that he/she purchased, from the ratings given to a specific item or from the item pages visited etc. The task of the item based algorithm is basically to find the likes and dislikes of the user and find items that will be liked by the user.

2.1.2.3. Implementation

While implementing the item based filtering various methods can be seen. According to Melville et al. [9] content based filtering can be seen as a text categorization problem. They tried to give a label to an item between 0 and 5 which corresponds to a user rating. During this process they have used a modified Naïve Bayesian text classifier to predict the label of the item given by the user by using previous votes of the user. Apart from this, other data mining and classification methods can be used such as decision trees, neural nets, vector-based representations and linear classifiers [5][11].

As can be seen from Figure 1.1 there is some sort of feedback mechanism employed to give better results in time. This method is generally referred to as relevance feedback [5]. The idea behind relevance feedback is giving the user a means for giving relevance information about the supplied recommendation so that using that feedback given recommendations may be tuned to give better results. Roccio's algorithm and probabilistic relevance feedback are the two algorithms used for this purpose [12].

2.1.3. Hybrid Algorithms

Both the collaborative approach and the content based approach have their strength and weaknesses which will be mentioned in the next section. So the third approach tries to combine the two methods in order to utilize their strengths and get rid of their weaknesses. The latest studies on recommender systems are one way or another in this form. One of the earliest studies on information filtering is made by Basu et al. [25]. The main idea of this study is that; combining elements of the social and content-based approaches makes it possible to achieve more accurate predictions.

There are different approaches while combining content-based and collaborative filtering techniques.

2.1.3.1 Linear Combination

The first approach is combining the results of content-based and collaborative recommendations linearly [27]. To apply this technique first the two techniques are used separately to achieve predictions and then some kind of weighted average of both results is used to give the final prediction [11]. The weights will determine which recommendation technique will be more dominant in the final recommendation (See Figure 2.1). Although weights can be predetermined an adaptive system can be used which will change the weights according to the sparseness of the dataset [28]. In the adaptive system the weight of the content based recommender results will be lowered accordingly as the ratings data gets richer. Apart from this different weighting systems may be used to achieve better results for datasets with different characteristics [10].

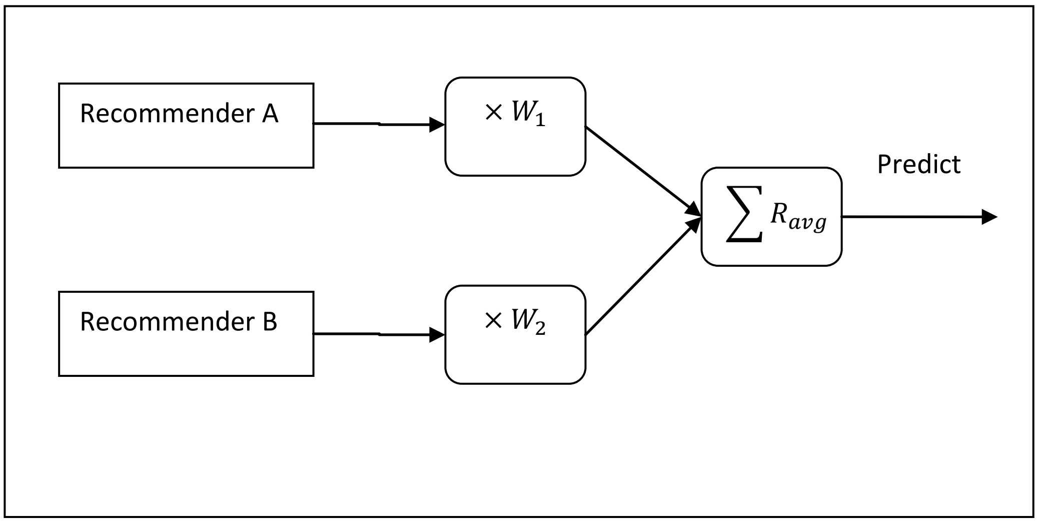


Figure 2.2. Linear Combination

2.1.3.2. Sequential Combination (Cascade)

In this technique first content based recommendation process is applied to find similar users and afterwards collaborative recommendation is done [27] or vice versa. Generally second stage is for items that the first stage cannot give a satisfying recommendation [11] and refines the recommendations of the first method. This method is more efficient then the weighted average stage because not all of the items in the dataset need to be processed by both of the techniques. Melville et al. uses this technique to give recommendations [10].

2.1.3.3. Clustering

Another method can be to use clustering the contents of the items in order to integrate them into collaborative filtering [27]. In the method mentioned in Li et al. these steps are used to give hybrid recommendations;

i. Apply the clustering algorithm to group the items and then create a group-rating matrix.
ii. Calculate the similarities of the group rating and item rating matrices separately and linearly combine the two results.
iii. Make a prediction for an item by performing a weighted average of deviations from the neighbors' mean.

2.2. Problems

Although the recommender systems researches have proved to be useful in a lot of cases still there are problems associated with different techniques. In this section we will talk about some of the common problems regarding recommender systems.

2.2.1. Cold-Start Problem

This is the case when a new user or item enters the scene. Since there is little information on the item/user recommender systems will have difficulty to make recommendations. In the case of a new user, aka first-rater, the collaborative filtering recommender cannot give recommendations to the user. In this case content information of the user can be used to give a recommendation if the user has created a profile about his/her preferences [10].

2.2.2. Sparsity

This problem depends on the fact that not all users rate all the items in the database. So when we are trying to give a recommendation for an item-user pair there may be little ratings that we can use to make a prediction which can lead to ineffective recommendations. For example in the dataset we used there are 6040 users, 3900 movies and ~1 Million votes. Each user has voted at least 20 movies. When we take the mean it can be seen that an average user has voted approximately 166 movies which corresponds to 4.26% of all movies which is a fairly large penetration ratio regarding real life datasets. In a sparse dataset it is hard to find similar users who have voted the same movies to find nearest neighbors.

One of the solutions proposed by Sarwar et al. [22] is to fill the user item matrix by either using the average ratings for a customer or average ratings for an item. It was found that product average give better results. They also propose to use dimensionality reduction of the sparse matrix with SVD. Another method is filling missing values in the item-user matrix with random values then use EM algorithm (see section 3) to increase the log-likelihood is a different method to overcome this method [31].

2.2.3. Overspecialization

Overspecialization is generally a problem for content-based recommenders. Since a recommender system recommends items that are similar to the items on the users profile recommendations are limited with those that are already rated. It's also a problem to recommend items that are very similar to those already seen because it is the duty of the recommender to make recommendations to the user that helps him/her to see/buy items that are different from the items on his/her profile [3].

To get rid of this problem there are different diversification methods [32]. These methods aim to add some kind of randomness to the recommendations so that the users can see diverse but somewhat relevant results. In their study Yu et al. they try to achieve explanation based diversification [32]. Here explanation refers to the reason behind the user preferring the item. They have defined the explanation for an item for a content-based system as the set of similar items that the user has liked in the past. A notion called explanation based diversity is introduced which is defined as the distance between two recommended items depending on their explanations. The goal is optimizing results with maximum relevancy and maximum diversity.

2.2.4. Scalability

As the number of items in the ratings dataset increases the computation requirements of a recommender system increases correspondingly. The problem related to this concept is called scalability. The recommender system must have the means to deal with this problem. An approach can be to cluster the items or users before applying collaborative filtering [27]. This will reduce the computation cost of the recommendation as the recommender will work only on the related cluster. A second method proposed is item based filtering technique where a precomputed model is formulated [16]. In this approach relationships between the items are explored and since these relationships are more static less online computing is required. Moreover Sarwar et al. [16] suggests that applying SVD matrix reduction model significantly increases scalability.

2.3. A Recommendation Case: Movie Recommendation

In real life there are lots of sites that employ recommendation techniques to give movie recommendations to their users. The CNET "Top 10 movie recommendation engines" list is a good resource [46]. Here we will try to give brief overviews of the most famous ones.

One of the most famous sites regarding movie recommendations is Netflix [45]. On the site on demand video streaming and DVD rental service are given to the subscribers. They give recommendations to their users using a collaborative filtering algorithm called Cinematch which is using version of Pearson correlation algorithm and multivariate regression to give recommendations [33]. In Netflix internet site it is mentioned that 60% of Netflix members select their movies based on movie recommendations. Netflix is most famous because of its 1 million $ prize that they offered in 2006 which will be given to the contestants who could beat their recommendation algorithm by a given percent [33] depending on the Root Mean Square Error (see section 4). They published a dataset of over 100 million votes given to the movies which is an anonymyzed version of their database. The dataset consisted of user IDs, movie IDs and ratings between 1 and 5. The Grand Prize was won by BellKor Team by Robert M. Bell, Yehuda Koren and Chris Volinsky on 2009. Their algorithm is a mixture of K-Nearest Neighbor, SVD, Restricted Boltzmann Machines and some other techniques [34].

The most well known site on movies is the Internet Movie Database (IMDb). It is one of the most used resources on the WWW by internet users before going to see a movie. According to Alexa IMDb is in the 42. place in usage regarding the global internet traffic [48]. In IMDb site they have nearly 1.7 million titles consisting of movies, tv shows, short films etc., 742,953 of them with recommendations. Users can have information on the items including genre, plot, summary, reviews, cast, synopsis and much more. The site offers recommendations giving similar items as the recently browsed item. They are using a complex algorithm that uses factors such as user votes, genre, title, and keywords to generate an automatic response [47].

One of the most interesting Movie Recommendation Engines is Jinni [49]. It may be the best example of a social recommendation engine because not only you can get recommendations depending on your ratings but also you can find people with similar taste. Their recommendations depend on a method called Movie Genome. They are trying to grab the "personality" of a movie building a Genome of the movie by taking into consideration some elements apart from the classic attributes; e.g. title, genre, cast; such as Experience, Story, structures, flags, awards and etc. At the beginning they have generated a gene pool by a team of movie professionals and tagged some movies using these genes. Now when a new film arrives they are indexed analyzing user reviews and metadata automatically. Each user has a "Movie Personality" which is constituted from his/her previous preferences and updated continuously. Jinni is more comprehensive, inclusive and innovative than most of the other systems. Jinni counts in different aspects of a movie to group movies. So a user can search for a movie for example regarding its mood whether it is a feel-good, atmospheric or humorous one, or can watch movies taking place in a specific country or era. In addition to this user can search similar movies. As an example when you search similar movies for "Jane Eyre" you get "Wuthering Heights", "Sense and Sensibility" and so forth, each displayed according to its relevancy to the target movie.

Our last example is the Movielens Movie Recommendation site with the motto of “helping you find the right movies” [50]. When someone first registers, in order to generate personalized movie recommendation the recommender engine generates a list of movies that the user will vote. The vote will be between 1 and 5 stars with the meanings as depicted in Figure 2.2.

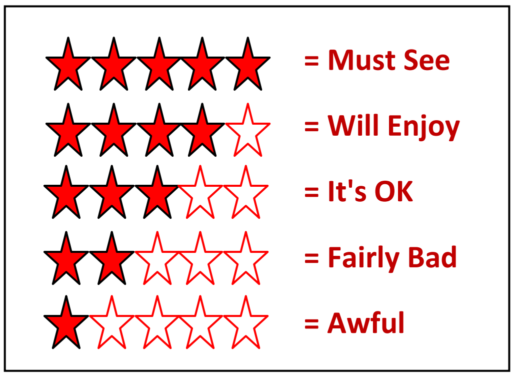
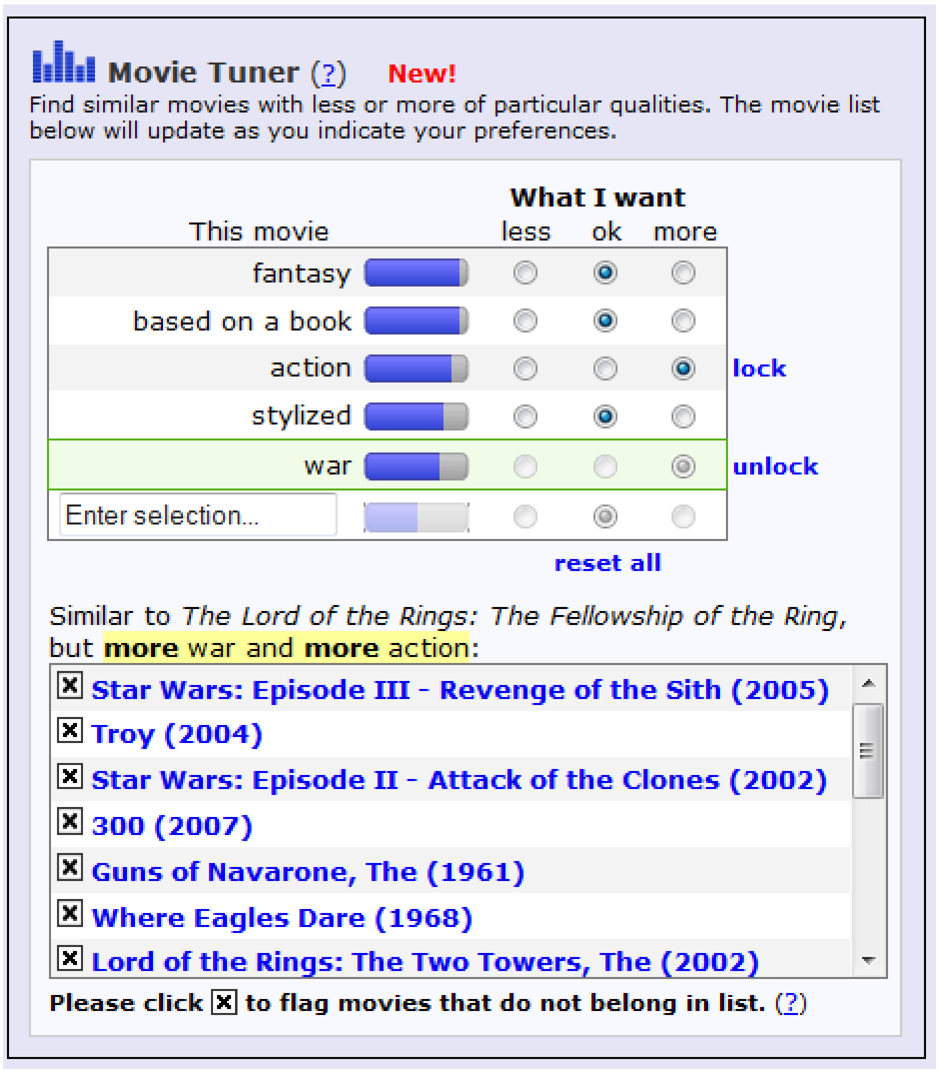


Figure 2.3. Movielens Ratings

Half star points can be given. User may not prefer to vote a movie he/she has not seen. Then a new set of movies come until 15 movies are rated. After rating 15 movies that were chosen from a wide range of movies personal movie recommendations for the user are given. In the site user can find a lot of different functions including picks for the user, wish list for the movies that has not yet been seen, find buddies that has similar taste, find information and statistics over movies and search for movies. The site also has a function called movie tuner which helps you find similar movies to the chosen movie but with some different attribute quantities. An example can be seen in Figure 2.3. Here the system tries to find movies similar to the famous fantasy movie “The Lord of the Rings: The Fellowship of the Ring” but with more action and more war content.



Movie Tuner (?) **New!**

Find similar movies with less or more of particular qualities. The movie list below will update as you indicate your preferences.

This movie		What I want: less	ok	more	
fantasy	▰	○	●	○	
based on a book	▰	○	●	○	
action	▰	○	○	●	lock
stylized	▰	○	●	○	
war	▰	○	○	●	unlock
Enter selection...	▱	○	●	○	

reset all

Similar to *The Lord of the Rings: The Fellowship of the Ring*, but **more** war and **more** action:

- ☒ Star Wars: Episode III - Revenge of the Sith (2005)
- ☒ Troy (2004)
- ☒ Star Wars: Episode II - Attack of the Clones (2002)
- ☒ 300 (2007)
- ☒ Guns of Navarone, The (1961)
- ☒ Where Eagles Dare (1968)
- ☒ Lord of the Rings: The Two Towers, The (2002)

Please click ☒ to flag movies that do not belong in list. (?)

Figure 2.4. Movielens Movie Tuner

CHAPTER 3

ENHANCED MOVIE RECOMMENDER SYSTEM

This chapter is devoted to our statistically enhanced recommender system. First part is an overview of the system. Then Expectation Maximization (EM) algorithm which is the method of choice for clustering is introduced. Afterwards the implementation details of the system are discussed.

3.1. Overview

Enhanced movie recommender system uses a statistical approach first to group users and movies in similar clusters then to find statistically relevant relations and patterns among users and user votes on movies so as to achieve a movie recommendation system both precise and scalable.

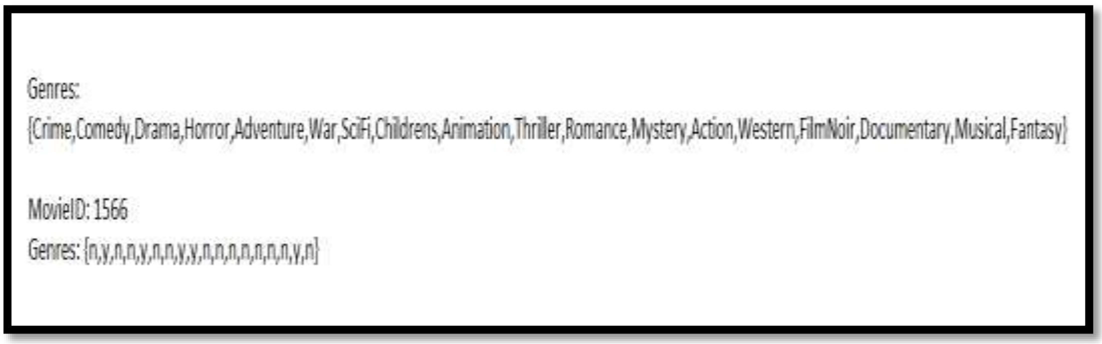
Our approach consists of these steps:

- Clustering
- Statistical Analyzing
- Rating Prediction

The next three subsections are about these steps.

3.1.1. Clustering

At first hand users and movies are clustered using Expectation Maximization Algorithm. First movies are clustered using genre information. As a movie can have more than one genre first we form a genre vector consisting of ‘y’s and ‘n’s. A ‘y’ means that the movie belongs to this genre and an ‘n’ means that the movie is out of genre. In the Figure 3.1 you can see genres space and a sample genre vector.



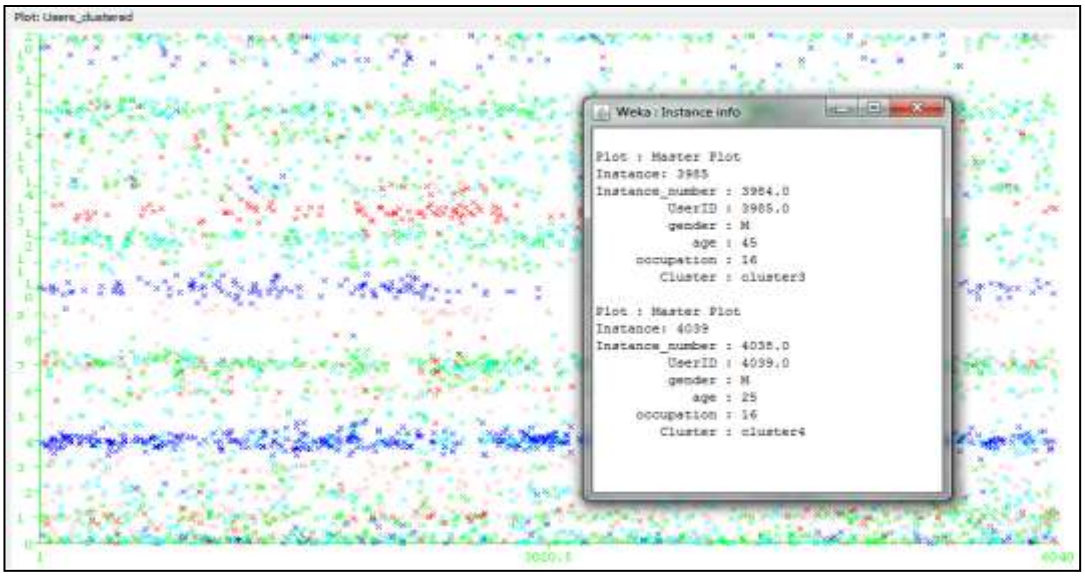
```
Genres:
{Crime,Comedy,Drama,Horror,Adventure,War,SciFi,Childrens,Animation,Thriller,Romance,Mystery,Action,Western,FilmNoir,Documentary,Musical,Fantasy}

MovieID: 1566
Genres: {n,y,n,n,y,n,n,y,y,n,n,n,n,n,n,n,y,n}
```

Figure 3.1. Genres

Second users are clustered first using their demographic data consisting of age, gender and occupation and than using their voting data on movie clusters. These clusters will be used to find statistically relevant users and movies and help finding results quickly.

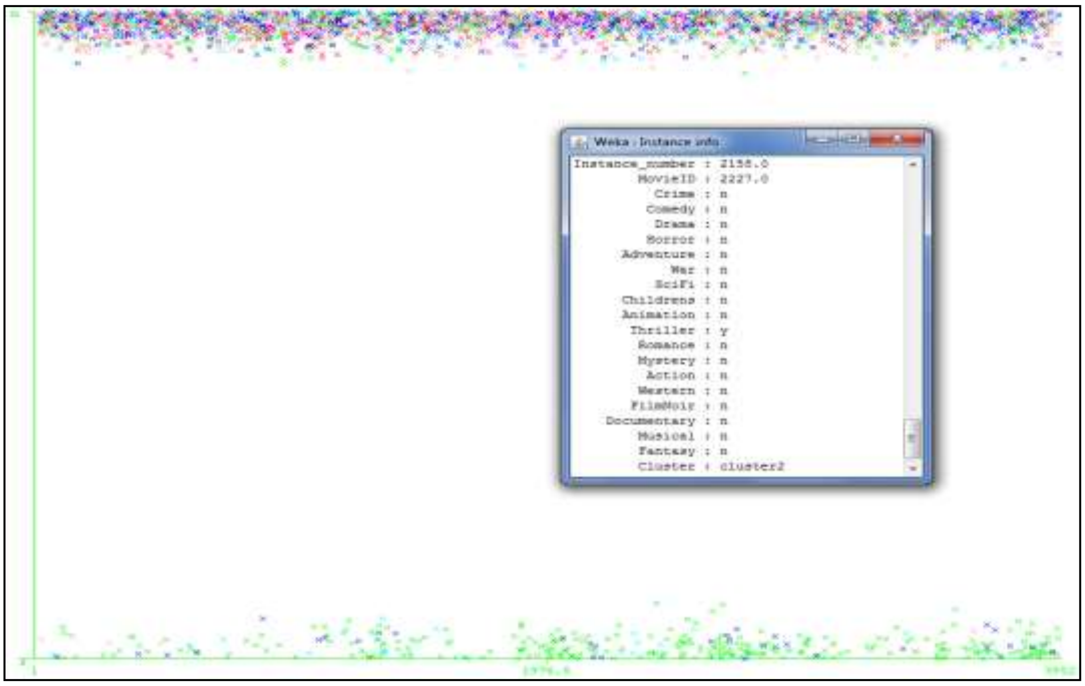
In Figure 3.2 you can see the user clusters according to demographic info over the occupation vs. UserID graph. There are 5 different clusters here which are shown in different colors. There is also the cluster information about two randomly selected users.

In Figure 3.2 you can see the clustering plot of movies with horror content. Details of the EM algorithm which is used for clustering is briefly introduced in the next section.



```
Plot : Master Plot
Instance: 3985
Instance_number : 3984.0
         UserID : 3985.0
         gender : M
            age : 45
     occupation : 16
        Cluster : cluster3

Plot : Master Plot
Instance: 4039
Instance_number : 4038.0
         UserID : 4039.0
         gender : M
            age : 25
     occupation : 16
        Cluster : cluster4
```

Figure 3.2. Sample Clusters



```
Instance_number : 2156.0
        MovieID : 2227.0
          Crime : n
         Comedy : n
          Drama : n
         Horror : n
      Adventure : n
            War : n
          SciFi : n
      Childrens : n
      Animation : n
       Thriller : y
        Romance : n
        Mystery : n
         Action : n
        Western : n
       FilmNoir : n
    Documentary : n
        Musical : n
        Fantasy : n
        Cluster : cluster2
```

Figure 3.3. Sample Movie Clusters

3.1.2. Statistical Analyzing

In this step items, users and votes are grouped in a statistical manner using [user, vote, item] matrices. This step is used to fine tune the clusters obtained from the previous step. In this step the underlying variables that can help predict how users vote an item is found out. A bunch of latent variables are to be considered and we tried to include the ones that makes more sense and adds a significant value. Recently there have been great deals of studies that tried to boost the performance of the system or increase the accuracy of the results and we benefited greatly from these works. Especially the studies made during the Netflix Grand Prize challenge and the Proceedings of the KDD Workshop on Large Scale Recommender Systems and the Netflix Prize [43] are very beneficial with regard to increasing the prediction accuracy.

3.1.3. Rating Prediction

In the last step of the system recommendations are given based on the previous results obtained from the clustering and analyzing processes and utilizing collaborative methods using basic nearest neighborhood algorithms.

At this stage we try to estimate a movie rating given to a specific movie by a specific user using the user's previous movie ratings and the votes of the similar users that belongs to the same cluster, which is found on the clustering phase, as the user. The clustering phase kept us from being overwhelmed with data that is unnecessary in perspective of relevance and while decreasing the processing time the prediction accuracy can compete with prediction methods without clustering.

3.2. Expectation Maximization and Clustering

Expectation maximization is a statistical algorithm used for maximum-likelihood estimation using incomplete data. The algorithm is first introduced in the fundamental paper of (FUND) and consists of an expectation step and then a maximization step. The mathematical background of the algorithm is a little overwhelming but there are some useful papers, presentations and books that aim to help user understand the fundamental idea behind the algorithm [38][35][46]. Here a brief overview of the algorithm and details on how it is used in clustering will be given.

3.2.1. Introductory Statistics

Before going to any further details some introduction to statistics is necessary. This section is far from being comprehensive, but it will be a brief introduction to the concepts we use along the thesis study.

For the following definitions assume a Random Variable X in a probability space$S$.

i. Cumulative Distribution Function

Cumulative distribution function gives the probability of a random variable X having a value $\mathrm{X} \leq \mathrm{x}_1$ that is;

$$\mathrm{P}(\mathrm{X} \leq \mathrm{x}_1) = \mathrm{D}(\mathrm{x}) \tag{3.1}$$

ii. Probability Density Function (pdf)

The probability density function $\mathrm{f}(\mathrm{x})$ of a random variable X is the derivative of the cumulative distribution function.

$$\mathrm{f}(\mathrm{x}) = \frac{\mathrm{d}}{\mathrm{dx}} \mathrm{D}(\mathrm{x}) \tag{3.2}$$

The two main properties of $\mathrm{f}(\mathrm{x})$ is;

$$\forall \mathrm{x}: \mathrm{f}(\mathrm{x}) \geq 0 \tag{3.3}$$

$$\int_{-\infty}^{\infty} \mathrm{f}(\mathrm{x}) \mathrm{dx} = 1 \tag{3.4}$$

Say we have the curve of the probability density function given in Figure 3.4.

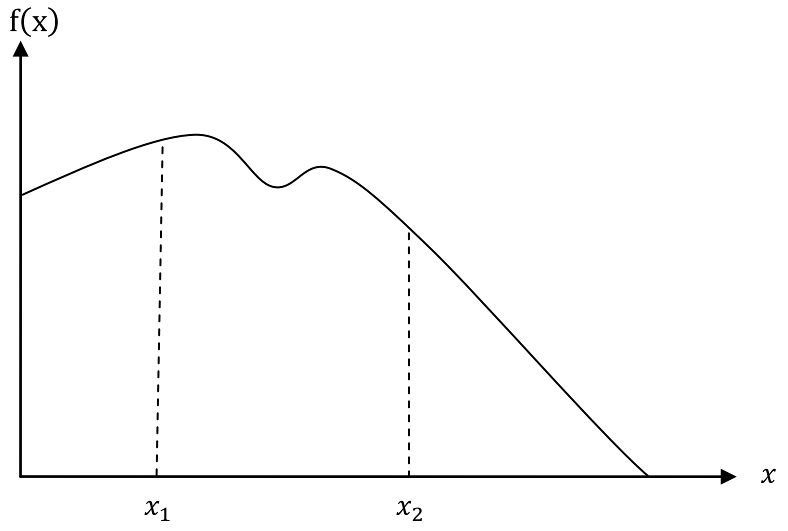


Figure 3.4. Probability Density Function

The probability of random variable X having a value $\mathrm{x}_1 \leq \mathrm{X} \leq \mathrm{x}_2$ is the value of the area under the $\mathrm{f(x)}$ curve between x_1 and x_2. In other words;

$$P(x_1 \leq X \leq x_2) = \int_{x_1}^{x_2} f(x)dx \tag{3.5}$$

iii. Expected Value (Mean)

Expected value (mean) ($\mathrm{E(x)}$) of a random variable is its average value over an infinite range of results, i.e.;

$$E(x) = \int_{-\infty}^{\infty} x\, f(x)dx \tag{3.6}$$

Mean is generally denoted by μ.

iv. Variance and Standard Deviation

Variance, $\mathrm{Var(x)}$ of a random variable can be calculated from;

$$Var(x) = E((x-\mu)^2) = \int_{-\infty}^{\infty} (x-\mu)^2\, f(x)dx \tag{3.7}$$

Standard deviation, σ, of a random variable is the square root of its variance.

$$\sigma = \sqrt{\mathrm{Var(x)}} \tag{3.8}$$

Hence variance is usually denoted as σ^2. Both variance and standard deviation is a measure of how far the probability distribution function is spread from the mean.

v. Mixture Model

Sometimes a probability density function $\mathrm{f(x)}$ of a random variable can be written as a combination of different probability density functions $\mathrm{f(x_i)} : i = \{1, \dots, n\}$ with different weights $\pi_i : i = \{1, \dots, n\}$.

$$f(x) = \sum_{i=1}^{n} \pi_i f(x_i) \tag{3.9}$$

This is called a mixture model. Generally the component densities $\mathrm{f(x_i)}$ belong to some parametric family, then $\mathrm{f(x)}$ can be written as

$$f(x) = \sum_{i=1}^{n} \pi_i f(x_i, \theta_i) \tag{3.10}$$

Where θ_i denotes the unknown parameter of the i$^{\text{th}}$ component density in the mixture [4].

3.2.2. Expectation Maximization (EM) Algorithm

The task of maximum likelihood estimation is to achieve a statistical model that best describes a set of data. The main task is to find the parameters regarding this statistical model using the data to for estimation. First a likelihood function for the given model over the given data is found. Then using some method like Newton-

Rhapson or EM model, parameters are estimated. We have chosen to use EM algorithm during the implementation of our recommender because it is easy to implement and stable [35].

EM algorithm, which is first introduced by Dempster et al. [35], is a generalized approach that can be used iteratively to compute maximum likelihood estimation. EM is particularly useful where data is incomplete. The algorithm is composed of an Expectation Step (E step) and a Maximization Step (M step).

The aim is of EM is to estimate some hidden or unknown parameter say θ which belongs to given data vector Y and some hidden variables ϑ which will help us find θ values easily. In the context of probability space we are trying to estimate the parameter θ so that the probability of $\mathrm{P}(\mathrm{Y}|\ \theta)$ is maximum.

The two main steps of the algorithm are as follows:

i. In the E Step posterior probabilities are computed for the latent variables ϑ based on the current estimates of the parameters
ii. In the M Step parameters are updated for the computed posterior probabilities.

The computations are processed iteratively until a satisfactory condition, e.g. a minimum change between iterations, are satisfied.

As described in [38] and [22] the log likelihood function, i.e., $L(\theta) = \ln \mathrm{P}(\mathrm{Y}|\ \theta)$ is typically used during implementation. Our aim is to maximize $L(\theta)$ iteratively. We can rewrite $\ln \mathrm{P}(\mathrm{Y}|\ \theta)$ as

$$\ln \mathrm{P}(\mathrm{Y}|\, \theta) = \sum_{\vartheta} \ln \mathrm{P}(\mathrm{Y}, \vartheta|\, \theta) \tag{3.11}$$

Given Y and θ', conditional expectation of $\ln \mathrm{P}(\mathrm{Y}, \vartheta|\, \theta)$ is;

$$\mathrm{Q}(\theta, \theta') = \mathrm{E}[\ln \mathrm{P}(\mathrm{Y}, \vartheta|\theta)|\mathrm{Y}, \theta] = \sum_{\vartheta} \mathrm{P}(\vartheta|\mathrm{Y}, \theta')\ln\mathrm{P}(\vartheta, \mathrm{Y}|\theta) \tag{3.12}$$

It can be proved that if

$$\sum_{\vartheta} \mathrm{P}(\vartheta|\mathrm{Y}, \theta')\ln\mathrm{P}(\vartheta, \mathrm{Y}|\theta) > \sum_{\vartheta} \mathrm{P}(\vartheta|\mathrm{Y}, \theta')\ln\mathrm{P}(\vartheta, \mathrm{Y}|\theta') \tag{3.13}$$

Then

$$\mathrm{P}(\mathrm{Y}|\, \theta) > P(Y|\, \theta') \tag{3.14}$$

So we can deduce that as long as we can improve expectation value of log likelihood using EM we can achieve better results.

Two main drawbacks of EM algorithm are that sometimes it is very slow to converge and it may reach a local maximum. There are various methods developed to overcome the slow convergence problems and it is advisable to run the algorithm beginning with different initial values to achieve an optimal maximum [37].

3.2.3. Clustering Using EM

Clustering is the task of partitioning a set of items into smaller group of similar items. There are various techniques used for data clustering. Some popular clustering algorithms are k-means, hierarchical clustering algorithm, Self Organizing Map Algorithm and Expectation Maximization clustering algorithm [42].

While some clustering algorithms assign each item a single cluster membership, e.g. k-means, some algorithms like EM assign each item a probabilistic cluster membership (soft clustering) [40].

The method of clustering for EM is such that; the algorithm tries to find dense regions of the probability density of the given data assuming that the probability density function can be modeled as a mixture of normal (Gaussian) distributions [41].

A Gaussian distribution for a vector X has a probability density function in the form of 3.15.

$$N(X, \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{x-\mu}{\sigma})^2} \tag{3.15}$$

A mixture of Gaussians can be written as in 3.16.

$$P(X) = \sum_i W_i N(X, \mu_i, \sigma_i) \tag{3.16}$$

W_i is the weight associated to each Gaussian distribution i in the mixture model depending on the dataset.

```
Require: number of cluster centers k
1: for all clusters do
2: randomly choose mean μ and covariance σ
3: end for
4: repeat
5: for all data points do
6: assign cluster membership probabilities to each data point
\\ Expectation Step
7: re-estimate Gaussian distribution parameters μ and σ
\\ Maximization Step
8: end for
9: until changes of parameter being lower than a threshold or
maximum number of iterations reached
10: return data points and their corresponding cluster
```

Figure 3.5. EM Clustering Algorithm

In Figure 3.5, the pseudo code of the algorithm for clustering with EM as appeared in [40] can be seen.

According to the algorithm first each cluster is assigned random W, μ, σ values. Here it is important to choose the values carefully so that the algorithm converges faster. To achieve this some can feed the results from a different clustering algorithm to EM for example k-means [40]. Then in the Expectation step the probability of each point x from data X belonging to a cluster C_i for all clusters 1, .., k is computed [41].

$$P(C|x) = \frac{WP(x|C)}{P(x)} \tag{3.17}$$

Then in the maximization step mixture model parameters are updated over the data X.

$$W' = \frac{1}{N}\sum_X P(C|x) \tag{3.18}$$

$$\mu' = \frac{\sum_X x.P(C|x)}{\sum_X P(C|x)} \tag{3.19}$$

$$\sigma' = \frac{\sum_X P(C|x)(x-\mu')(x-\mu')^T}{\sum_X P(C|x)} \tag{3.20}$$

The iterations continue until the log-likelihood of the mixture model converges.

$$\ln(X, \mu, \sigma, W) = \sum_X \ln\left(\sum_{i=1}^{k} W_i N(x, \mu_i, \sigma_i)\right) \tag{3.21}$$

3.2.4. System Details

Throughout the implementation phase the main development environment used is the Visual Studio 2010 platform from Microsoft. The programming language of choice is C#.Net (C-sharp). The reason of choice is being readily acquainted with the aforementioned platform and the .Net programming environment having a lot of built-in library functions that makes the work of the programmer easier. Also I have used SQL Server 2008 Express Edition database management system which comes with Visual Studio 2010. This version of SQL server is sufficient for development purposes.

The data mining and statistical analyzing tool used is a Java programming language based tool called Weka from Machine Learning Group at University of Waikato [44]. Weka can be used by means of the graphical user interface (GUI) or can be called directly from within Java Code. It has many built in tools for data pre-processing, classification, regression, clustering, association rules, and visualization.

The test machine used during the processing phase has a 64 bit Windows 7 Operating System working on an Intel Core 2 duo CPU with 2.40 GHz clock speed, 4 GB of Ram and 250 GB of 7200 rpm hard disk.

To test our algorithms we have chosen to use is one of the Movilens movie data sets from GroupLens [52]. The dataset we used has 6040 users, ~3900 movies and ~1 Million votes. The minimum number of ratings for a user is 20, maximum number is 2314, the mean for the ratings counts is 166 and the median is 96. The dataset consists of three text files which are called movie, user, and rating. User text file includes demographic data for users including age, gender and occupation. Age attribute consists of 7 different classes, each class including a different range of ages which are:

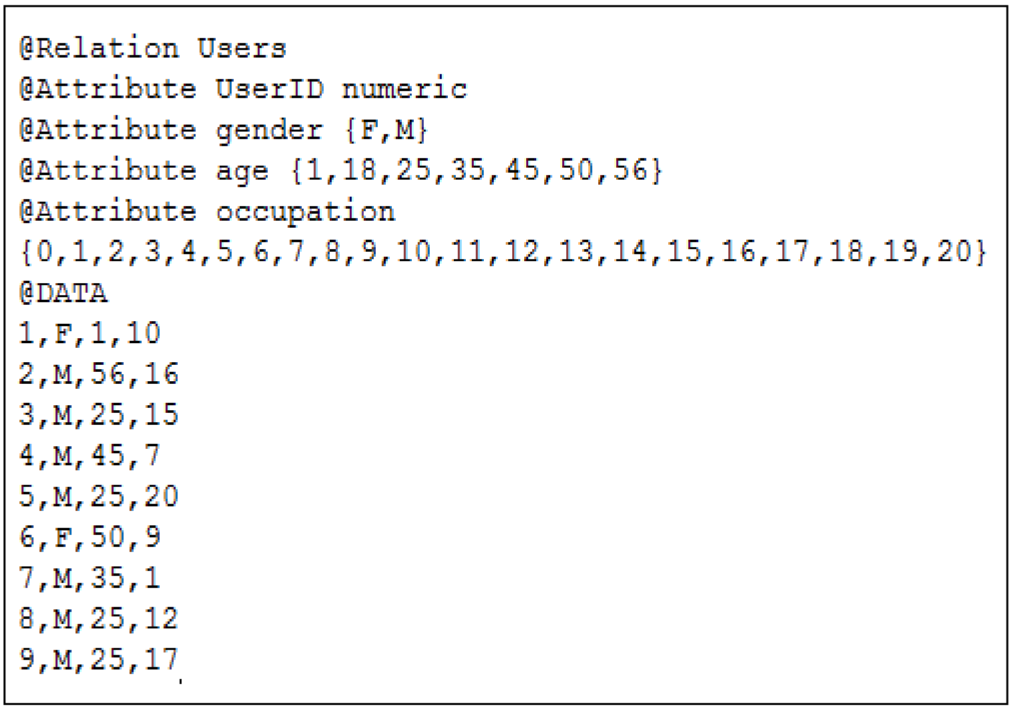
- 1: "Under 18"
- 18: "18-24"
- 25: "25-34"
- 35: "35-44"
- 45: "45-49"
- 50: "50-55"
- 56: "56+"

Occupation attribute is an integer between 0 and 20. Each integer is a different occupation and 0 meaning not specified. Movie text file includes genre information of movies which are chosen from 18 different genre types (See Figure 3.1). Each movie genre consists of the combination of genres chosen from this genre types.

In the design phase of our system we planned the system to have five steps of implementation. These are:

i. Preprocessing
ii. Clustering
iii. Statistical Analyzing
iv. Rating Prediction
v. Recommendation

In the preprocessing phase the Movielens dataset is first separated into a test set and a training set. The test set is chosen randomly to include 10% of all the data. Then the data is processed to be compatible with the Weka “arff”(Attribute-Relation File Format) file [50]. An arff file has two distinct sections. These are Header and Data sections. The **Header** of the ARFF file contains the name of the relation, a list of the attributes (the columns in the data), and their types. A sample from one of the arff files used in the implementation is in Figure 3.6.



```
@Relation Users
@Attribute UserID numeric
@Attribute gender {F,M}
@Attribute age {1,18,25,35,45,50,56}
@Attribute occupation
{0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20}
@DATA
1,F,1,10
2,M,56,16
3,M,25,15
4,M,45,7
5,M,25,20
6,F,50,9
7,M,35,1
8,M,25,12
9,M,25,17
```

Figure 3.6. Sample arff file

The clustering phase is done with the Weka GUI working on the arff files created within the C-sharp program written. The method of choice is the EM clustering algorithm. While using EM in WEKA the number of clusters to be formed can be given, if not WEKA uses cross validation to automatically determine the number of clusters. Steps of the cross validation performed to determine the number of clusters is;

i. the number of clusters is set to 1

ii. the training set is split randomly into 10 folds.

iii. EM is performed 10 times using the 10 folds the usual CV way.

iv. the loglikelihood is averaged over all 10 results.

v. if loglikelihood has increased the number of clusters is increased by 1 and the program continues at step 2.

In our method we use three levels of clustering. First the users are clustering according to demographic data, second the movies are clustered according to genres and third the users are clustered according to the given ratings to the movie clusters formed to cluster the users according to their “movie taste”.

In the phase of statistical analysis phase we have done various analyses on the rating data. We looked for latent factors that affect the given rating from simple such as occupation or being a male or female to more complex ones. The rating habits of users such as being a high or low voter, correlation of the votes to the general mean or not, content preferences of the user, being a regular voter or not etc. are also inspected. Statistics is a good resource for recommendation since it can help us find hidden factors and patterns of voting which may help us give more accurate and relevant recommendations reflecting the users’ preferences.

After the clustering phase the collaborative filtering methods mentioned in the second chapter are used to give a numerical prediction between 1 and 5. This prediction will help us both to find movies that will be best liked by the user and to evaluate the accuracy of our recommender system in the following chapter. We use a weighting scheme that will change the weights of the content-based and collaborative parts of our recommender to adapt for the users who have scarce ratings.

At the last phase recommendations, depending on the rating predictions done in the previous steps, are given to the user via an XML file. Choosing an XML file is to have a generic framework that can be used to be shown through a graphical user interface or a web page or can be used through a web service. When an interactive system is implemented in the future these XML files can be updated easily as new data is obtained from the users.

CHAPTER 4

EVALUATION

This chapter of the study is dedicated to the evaluation of the recommendation performance of the Enhanced Movie Recommender System. In the first section the methods and metrics that are used for performance measurement of Recommender Systems are introduced. In the second section the test results of the Enhanced Movie Recommender are given.

4.1. Evaluation Criteria

There are several metrics that have been used in different studies. In general these metrics can be divided into three groups which are Statistical Accuracy Metrics, Coverage Metrics and Decision Support Accuracy Metrics [39].

4.1.1 Statistical Accuracy Metrics

These metrics measure the overlapping of the predicted numerical ratings and the actual ratings. The two main methods used for statistical accuracy measurement are Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) [26].

i. Mean Absolute Error (MAE)

MAE is the measure of the difference between the predicted ratings and the actual ratings [1]. Assume a movies vector $\{m_1, m_2, \ldots, m_n\}$ consisting of n movies and corresponding predicted ratings vector $\{p_1, p_2, \ldots, p_n\}$ and actual ratings vector $\{r_1, r_2, \ldots, r_n\}$. MAE can be calculated as:

$$MAE = \sum_{i=1}^{n} \frac{|p_i - r_i|}{n} \tag{4.1}$$

As the MAE decreases prediction accuracy increases.

ii. Root Mean Square Error (RMSE)

RMSE is a different kind of metric regarding the absolute difference between the prediction vector and the ratings vector. Using the same variables as in the MAE case RMSE can be calculated as:

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{|p_i - r_i|^2}{n}} \tag{4.2}$$

Again smaller RMSE implies better prediction accuracy.

4.1.2. Coverage Metrics

Coverage is a metric showing the percentage of the items in the item vector that the system can provide recommendations. Coverage (Cov) can be calculated by taking the percentage of the items that has ever been recommended (I_R) of all the items in the dataset (I_T) that is

$$Cov = \frac{I_R}{I_T} \times 100 \quad (4.3)$$

The coverage percent and the relevance of the recommendations are generally has an inverse relationship so these values must be optimized [29].

4.1.3. Other Metrics

There are also other kind of metrics that can be used for recommender system evaluation, but we have not used in our study. One kind of metrics used is Decision Support Accuracy Metrics (DSAM) which is interested in what degree the recommendations help the users to choose a movie that they are interested in. From the point of view of DSAM the recommendation made on an item is either good or bad [22]. There are different kinds of DSAM which are Reversal Rate and Receiver Operation Characteristic (ROC) Sensitivity. Other than DSAM there are different metrics called recall, precision, diversity and novelty, learning rate and confidence [15][29].

4.2. Test Results

We have used both the 100K ratings dataset and the 1M ratings dataset obtained from Movielens. We have tested general prediction accuracy using 100K dataset. To evaluate our algorithm against new user and new item problems we have used 1M dataset. For this purpose first we have separated 20 random users and 20 random movies from the training set. Then we have separated 10% of the remaining ratings as a test set. We then ran a various set of tests to evaluate the prediction accuracy, coverage against different number of clusters.

4.2.1. Prediction Accuracy

First using our algorithm we tried to give numerical predictions to some of the <user,movie> pairs in our test data set. Using these predictions we tried to find the prediction accuracy of our algorithm. During this test we use MAE of our predictions. We have tested our algorithm for 4 to 8 movie clusters; i.e. 4G, 6G and 8G; and corresponding interest groups ranging from 4 to 10.

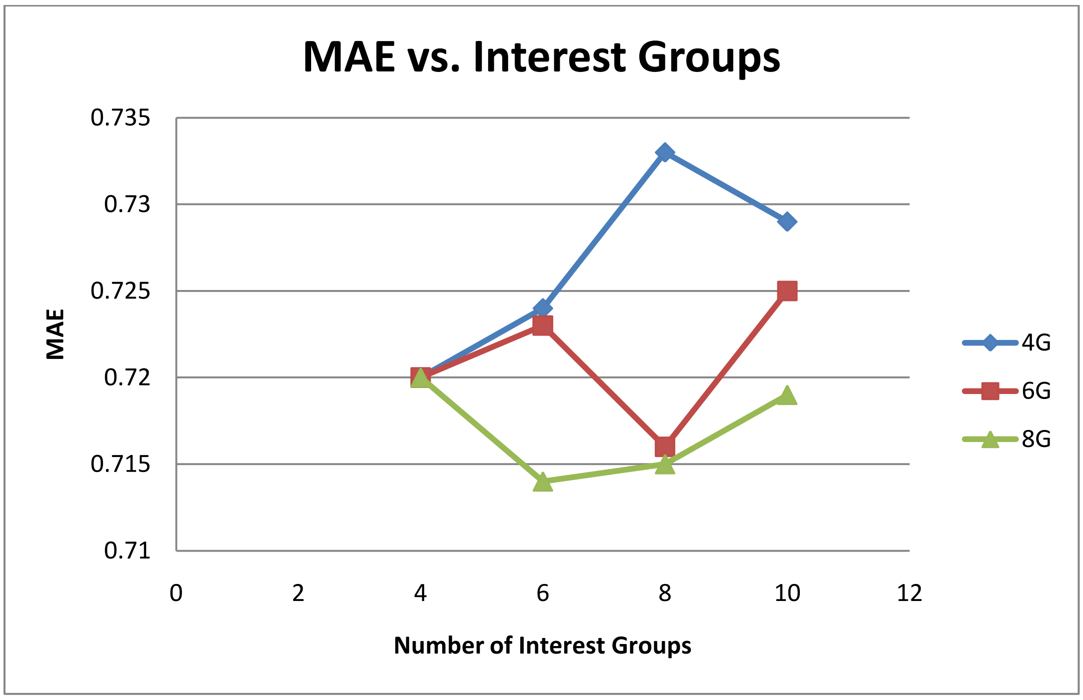


Figure 4.1. MAE vs. Interest Groups

As can be seen from Figure 4.1, increasing number of interest groups has a slight decreasing affect on prediction accuracy. But the accuracy is still reasonably high.

For comparison we use the two different test results taken from Sarwar et al. [14]. The two different results belong to the classical collaborative filtering (Classical) and the clustering approach used (Clust) in the aforementioned study. The comparison results can be seen in Table 4.1.

Table 4.1. MAE for Different Methods

Method Used	Range of MAE
Classical	0.74-0.80
Clust	0.77-0.82
Enhanced	0.72-0.74

As can be seen from the given error rates the accuracy of Enhanced Recommender algorithm is good compared to the error rates of the classical collaborative filtering approach and the clustering approach used in Sarwar et al. [14].

4.2.2. Coverage

As long as coverage is concerned our algorithm has %100 coverage. This result arises from the fact that when there is not enough neighbors detected to give predictions for the current <user, movie> pair our algorithm gives a prediction that is has a linear relationship with the average rating value for the movie. Our algorithm finds the correlation the general user ratings with the global average of all users and then multiplies the average rating value for the movie with a weight and gives this result as a prediction.

It is possible to achieve better results by narrowing the coverage. We have tested the affect of changing coverage value to the prediction accuracy of our. For this test scenario we abandoned using average ratings and tested the system performance by changing minimum correlation value for neighborhood formation. That is the minimum correlation required for a user that is in the same cluster to the target user so that he/she will be considered a neighbor. We have changed minimum required correlation value between 0.7 and 0.3 to achieve the results in Figure 4.2.

The results show that while we can achieve a MAE 0.61 for pure correlation based recommendation with minimum correlation value of 0.7 the coverage decreases to 12.5% which is very low. For high coverage above 90% the MAE value rises up to 0.75. It is a challenge to find an optimum value of coverage that will provide a reasonable prediction accuracy and user satisfaction.

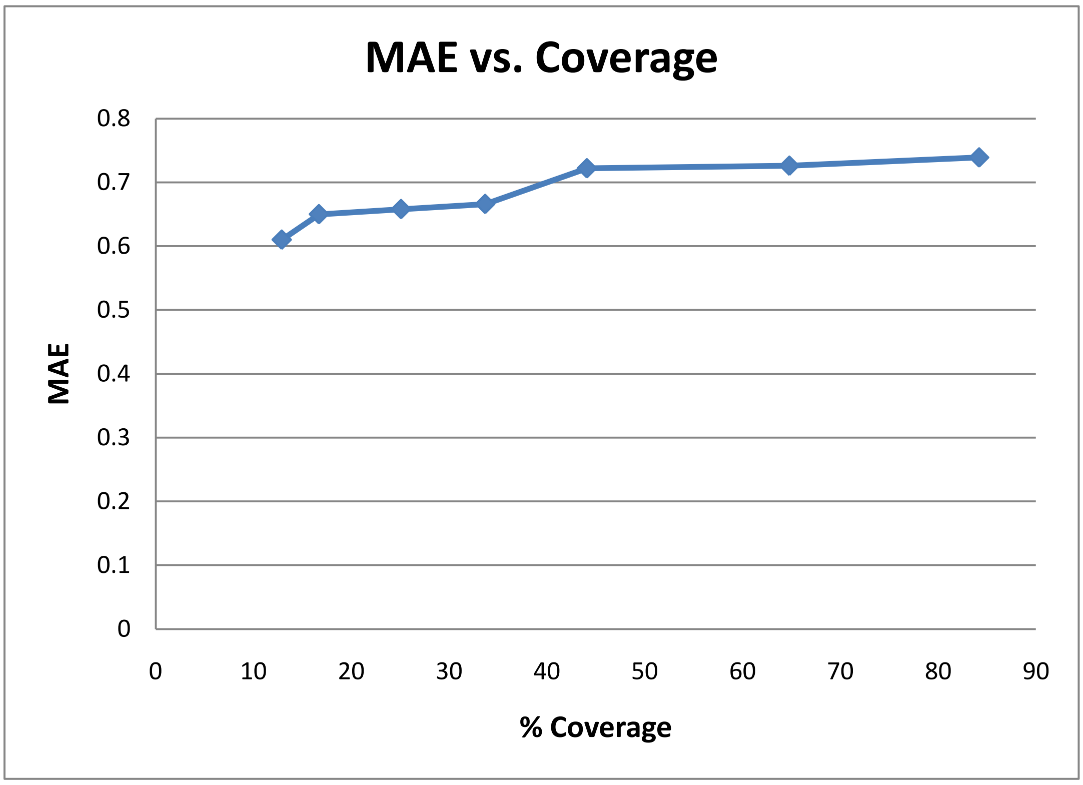


Figure 4.2. MAE vs. Coverage

4.2.3. New User

We tried to use the results from the demographic clustering phase in order to get rid of the new user problem and give good recommendation results to a newly arrived user with no initial rating data. During clustering we have neglected the users who have not stated their occupation. We have operated EM algorithm for different seed values to achieve an optimum result as it is known that EM doesn't guarantee an optimal maximum [37]. At the end a result with 5 clusters proved to give good results. In order to achieve a baseline for comparison we have evaluated the average recommendation case and random clusters case. In random clusters case we have formed 5 random clusters and evaluated the prediction accuracies.

In Table 4.2. the results that compare demographic recommendation to the average recommendation and random clustering can be seen.

Table 4.2. MAE for New User

1000 samples	Mean Absolute Error	Root Mean Squared Error
Random	0.756	0.944
Mean	0.757	0.945
Enhanced Recommender	0.747	0.932

Looking at the average error results it can be seen that our algorithm performs slightly better than the average recommendation case. Also the random recommendation case performs nearly equivalent to the average recommendation case. In addition for 1000 samples while average recommendation takes 25 seconds to process, Enhanced Recommender only needs 13 seconds. So we can conclude that Enhanced Recommender is better in terms of both accuracy and processing time compared to using the average rating results for recommending movies to a new user.

4.2.4. New Item

To find out how our system performs against a new movie arriving in the database we tried to predict ratings that will be given to the movie test set that we have separated from the movie set. In this step we used the clustering results of the movies according to genre. We applied EM clustering algorithm several times and tried to find the optimum number of clusters. In Figure 4.3 you can see the effect of changing number of clusters to the prediction accuracy.

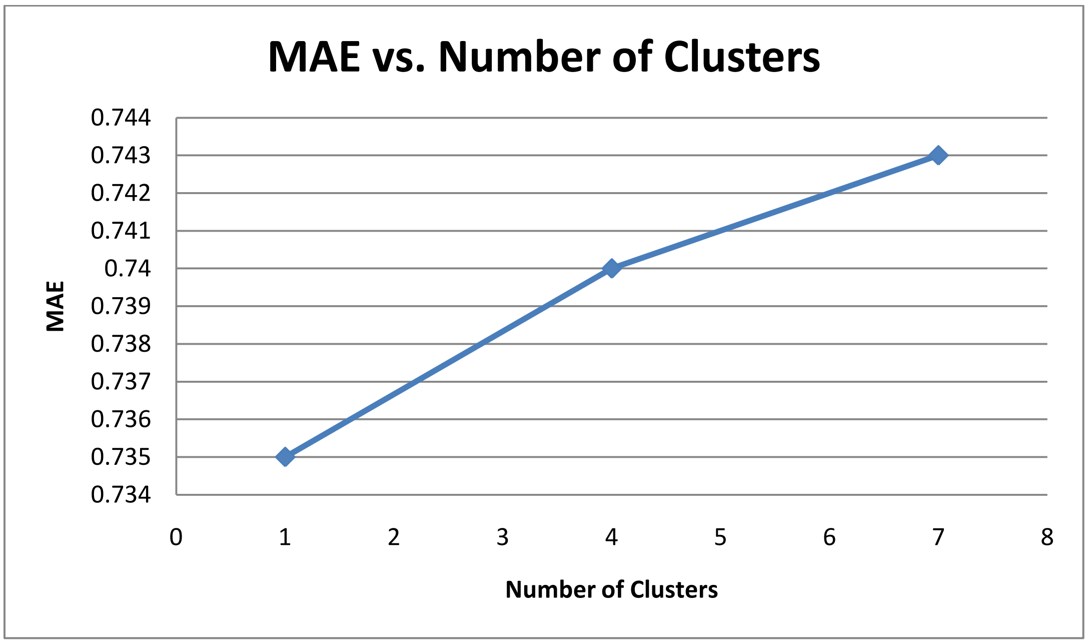


Figure 4.3. MAE vs. Number of Clusters

From these results it can be concluded that increasing number of clusters slightly decreases prediction accuracy. However the computation times for large number of clusters are smaller. Here there is a tradeoff between scalability and accuracy.

CHAPTER 5

CONCLUSION

Recommender systems are used to give reasonable advices to users on items that have a high probability to be liked or purchased. Usually the past purchases or ratings on items of the user and information on the user profile are utilized in the recommendation process. There are now many web sites and applications that make use of recommender systems to aid their users to find relevant and interesting items.

Recommender systems are also the subject of many researches done in the area of data mining. While there are a lot of research done on recommender systems, problems that need attention still exists. One of the problems yet to be solved is scalability. The main object of this study is to obtain recommendation results in a smaller time while preserving good prediction accuracy. The famous Expectation Maximization Algorithm, which is in fact a statistical method used for maximum likelihood estimation and predicting hidden variables, is used to implement a clustered, scalable, hybrid recommender system.

The main method used to achieve scalability is clustering. The method used follows a similar path as the one used in Sarwar et al. [14]. In their study Sarwar et al. used a variant of K-means clustering algorithm called bisecting K-means clustering algorithm to partition the user space in to smaller but related groups.

After clustering phase, neighborhood formation is done on the cluster that the target user belongs so that less computation is needed. The approach used in this study differs from Sarwar et al. [14] in two aspects. First EM clustering algorithm is used instead of K-means clustering algorithm. Second clustering is done in both user level and item level to form interest groups. Comparison of prediction accuracies of this study and the aforementioned study shows that Enhanced Movie Recommender System has a reasonably high accuracy. Also from the experiments over the data set of choice, it is found that the computation time needed for recommendation using the Enhanced Movie Recommender System is 2 to 3 times smaller than the classical collaborative filtering algorithm.

Second aspect of the Enhanced Movie Recommender system is statistical enhancements. Using statistical analysis methods the underlying variables that can help predict how users vote an item is found out. These results are used to increase the prediction accuracy of the system. In most of the systems employing clustering there is a tradeoff between prediction time and recommendation quality [17]. From the results of the evaluation phase (see chapter 4), it can be deduced that the statistical analysis phase has overcome the problem of accuracy degradation.

The Enhanced Movie Recommender System is tested for new user and new item problems which is the case when a new user or a new item enters the scene. For this case the content information for an item and the profile page of a user can be used to give recommendations. According to the test results the prediction accuracy of the system is fairly good.

5.1. Future Work

In the future some adjustments and enhancements to the algorithm will be made so as to increase prediction accuracy. The algorithm can be tested on some other dataset and it can be seen if the performance changes. The main aim of a recommender system is to supply users a better way to find interesting items so it will be reasonable to integrate Enhanced Movie Recommender System to a webpage with a graphical user interface and improve its algorithm using the feedbacks coming from the users.

REFERENCES

[1] **Shardanand, U., Maes, P.** (1995), *Social Information Filtering: Algorithms for Automating "Word Of Mouth"*, Proceedings of the Conference on Human Factors in Computing Systems, (CHI'95), Denver, CO, ACM, 210-217.

[2] **Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., Riedl, J.** (1994), *Grouplens: An Open Architecture for Collaborative Filtering of Netnews*, Proceedings of Conference on Computer Supported Cooperative Work (CSCW'94), Chapel Hill, NC, ACM Press, 175-86.

[3] **Adomavicius G., and Tuzhilin A.** (2005), *Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions*, In IEEE Transactions on Knowledge and Data Engineering, Vol 17, No. 6.

[4] **McLachlan, G., Peel, D.** (2001), *Finite Mixture Models*, John Wiley & Sons.

[5] **Pazzani, M.J., Billsus, D.** (2007), *Content-based Recommendation Systems*, in Brusilovsky, P., Kobsa, A., Nejdl, W., eds.: The Adaptive Web: Methods and Strategies of Web Personalization, Volume this volume of Lecture Notes in Computer Science, Springer Verlag.

[6] **Goldberg D., Nichols D., Oki B. M., Terry, D.** (1992), *Using Collaborative Filtering to Weave an Information Tapestry*, Communications of the ACM, 35(12):61-70.

[7] **van Meteren, R., van Someren, M.** (2000), *Using Content-Based Filtering for Recommendation*, Working paper, University of Amsterdam, Netherlands.

[8] **Salton, G., McGill, M.J.** (1983), *Introduction to Modern Information Retrieval*, McGraw Hill Book Co., New York.

[9] **Larocca, N. J., Santos, A.D., Kaestner, C.A.A., Freitas, A.A.** (2000), *Document Clustering and Text Summarization*, in Proc. 4th Int. Conf. Practical Applications of Knowledge Discovery and Data Mining, 41–55.

[10] **Melville, P., Mooney, R. J., Nagarajan, R.** (2002), *Content-boosted Collaborative Filtering for Improved Recommendations*, in Proc. 18th Nat. Conf. Artificial Intelligence, vol. AAAI-2002, pp. 187–192.

[11] **BURKE, R.** (2002), *Hybrid Recommender Systems: Survey and Experiments*, User Model, User-Adapt. Interact. 12, 4, 331–370.

[12] **Manning, C. D., Raghavan, P. Schütze, H.** (2008), *Introduction to Information Retrieval*, Cambridge University Press.

[13] **Breese, J. S., Heckerman, D., Kardie, C.** (1998), *Empiricial Analysis of Predictive Algorithms for Collaborative Filtering*, In Proceedings of the 14th Conference on Uncertainity in Aritificial Intelligence, 43–52.

[14] **Sarwar, B., Karypis, G., Konstan, J., Riedl, J.** (2002), *Recommender Systems for Large-Scale E-Commerce: Scalable Neighborhood Formation Using Clustering*, in Fifth International Conference on Computer and Information Technology.

[15] **Schafer, J., Frankowski, D., Herlocker, J., Sen, S.** (2007), *Collaborative Filtering Recommender Systems*, Lecture Notes in Computer Science, 4321:291.

[16] **Sarwar, B. M., Karypis, G., Konstan, J. A., Reidl, J.** (2001), *Item-based Collaborative Filtering Recommendation Algorithms*, in Proceedings of the World Wide Web Conference (WWWC10), 285–295.

[17] **Su, X., Khoshgoftaar, T.** (2009), *A survey of collaborative filtering techniques*, Advances in Artificial Intelligence.

[18] **Friedman, N., Geiger, D., Goldszmidt, M.** (1997), *Bayesian Network Classifier,* Machine Learning, 29(2/3):131–163.

[19] **Miyahara, K., Pazzani, M. J.** (2002), *Improvement of Collaborative Filtering with the Simple Bayesian Classifier*, Information Processing Society of Japan, vol. 43, no. 11.

[20] **Bennett, J., Eklan, C., Liu, B., Smyth, P., Tikk, D.** (2007), *KDD Cup and Workshop 2007*, ACM SIGKDD Explorations Newsletter, 9(2), 51-52.

[21] **Koren, Y., Bell, R., Volinsky, C.** (2009), *Matrix Factorization Techniques for Recommender Systems*, Computer 42(8), 30–37.

[22] **Sarwar B., Karypis G., Konstan J., Riedl J.** (2002), *Incremental Singular Value Decomposition Algorithms for Highly Scalable Recommender Systems*, In Fifth International Conference on Computer and Information Technology.

[23] **Collins, M.** (1997), *The EM Algorithm*, University of Pennsylvania.

[24] **Deerwester, S., Dumais, S., Furnas, G., Landauer, T., Harshman, R.** (1990), *Indexing by Latent Semantic Analysis*, Journal of the American Society for Information Science, 41(6), 391–407.

[25] **Basu, C., Hirsh, H., Cohen, W.W.** (1998), *Recommendation as Classification: Using Social and Content-Based Information in Recommendation*, in Proceedings of the Fifteenth National Conference on Artificial Intelligence, pages 714–720, Madison, WI.

[26] **Good, N., Schafer, J. B., Konstan, J. A., Borchers, A., Sarwar, B., Herlocker, J., Riedl, J.** (1999), *Combining Collaborative Filtering with Personal Agents for Better Recommendations*, in Proceedings of the AAAI'99, 439-446.

[27] **Li, Q., Kim, B.** (2003), A*n Approach for Combining Content-Based and Collaborative Filters*, in Proceedings of IRAL2003.

[28] **Claypool, M., Gokhale, A., Miranda, T., Murnikov, P.,Netes, D., Sartin, M.** (1999), *Combining Content Based and Collaborative Filters in an Online Newspaper*, In Proc. ACM-SIGIR Workshop on Recommender Systems: Algorithms and Evaluation.

[29] **Herlocker, J.L., Konstan, J.A., Terveen, L.G., Riedl, J.T.** (2004), *Evaluating Collaborative Filtering Recommender Systems*, ACM Trans. Information Systems, vol. 22, no. 1, 5-53.

[30] **Balabanovic, M., Shoham, Y.** (1997), *Fab: Content Based, Collaborative Recommendation*, Communications of the ACM, 40(3), 66–70.

[31] **Srebro, N., Jaakkola, T.** (2003), *Weighted Low-Rank Approximations*, in Proceedings of the Twentieth International Conference on Machine Learning.

[32] **Yu, C., Lakshmanan, L., Amer-Yahia, S.** (2009), *Recommendation Diversification Using Explanations*, in ICDE.

[33] **Bennet, J., Lanning, S.** (2007), T*he Netflix Prize*, KDD Cup and Workshop.

[34] **Bell, R.M., Koren, Y., Volinsky, C.** (2008), *The Bellkor 2008 Solution to the Netflix Prize*, Technical Report.

[35] **Dempster, A. P., Laird, N. M., Rubin, D. B.** (1977), *Maximum Likelihood From Incomplete Data Via The EM Algorithm*, Journal of the Royal Statistical Society, Series B, vol. 39, 1–38.

[36] **Xu, G., Zhang Y., Zhou, X.** (2005), *A Web Recommendation Technique Based on Probabilistic Latent Semantic Analysis*, in Proceeding of 6th International Conference of Web Information System Engineering, p. 15-28, New York City, USA.

[37] **McLachlan, G. J., Krishnan, T.** (1996), *The EM Algorithm And Extensions*, New York: Wiley.

[38] **Weinstein, E.** (2006), *Expectation-Maximization Algorithm and Applications*, Courant Institute of Mathematical Sciences.

[39] **Herlocker, J., Konstan, J., Borchers, A., Riedl, J.** (1999), *An algorithmic Framework for Performing Collaborative Filtering*, in SIGIR ’99: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 230–237.

[40] **Planinc, R.** (2010), *Modeling Sources and Sinks in Crowded Scenes by Clustering Trajectory Points Obtained by Video-based Particle Advection*, Thesis, Austrian Institute of Technology.

[41] **Bradley, P. S., Fayyad, U., Reina, C.** (1998), *Scaling Clustering Algorithms to Large Databases*, Proc. 4th Intl. Conf. on Knowledge Discovery and Data Mining (KDD98), AAAI Press.

[42] **Abbas, O. A.** (2007), *Comparison between Data Clustering Algorithms*, Yarmouk University, Jordan.

[43] **KDD** (2008), *Proceedings of the Second KDD Workshop on Large Scale Recommender Systems and the Netflix Prize*, Las Vegas, Nevada, USA.

[44] **Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I. H.** (2009), *The WEKA Data Mining Software: An Update*, SIGKDD Explorations, Volume 11, Issue 1.

[45] http://www.netflix.com, Retrieved November 12, 2010.

[46] http://news.cnet.com/8301-17939_109-10200031-2.html, Retrieved November 19, 2010.

[47] http://www.imdb.com, Retrieved November 11, 2010.

[48] http://www.alexa.com/siteinfo/imdb.com, Retrieved December 7, 2010.

[49] http://www.jinni.com, Retrieved December 2, 2010.

[50] http://weka.wikispaces.com, Retrieved October 13, 2010.

[51] http://movielens.umn.edu, Retrieved September 19, 2010.

[52] http://www.grouplens.org, Retrieved September 17, 2010.

APPENDIX A

CURRICULUM VITAE

PERSONAL INFORMATION

Surname, Name: ÖZKAN, Hüseyin Burhan

Nationality: Turkish (TC)

Date and Place of Birth: 13 October 1982 , Mersin

Marital Status: Single

Phone: +90 312 294 71 46

Fax: +90 312 294 71 73

email: hbozkan@btk.gov.tr

EDUCATION

Degree	**Institution**	**Year of Graduation**
BS	METU Electrical and Electronics Engineering	2007
High School	Yusuf Kalkavan Anadolu High School	2000

WORK EXPERIENCE

Year	**Place**	**Enrollment**
2008-Present	BTK	Assistant IT Expert
2007-2008	HAVELSAN	System Engineer
2007	ILG Studios	Intern as a Software Developer

FOREIGN LANGUAGES

Advanced English

HOBBIES

Books, Internet, Wing Tzun