MoresysGOAL: MOVIE RECOMMENDATION SYSTEM USING COLLABORATIVE FILTERING TECHNIQUE SUPPLEMENTED BY CONTENT WITH GOAL PROGRAMMING

A Thesis Submitted to
The Graduate School of Engineering and Sciences of
İzmir Institute of Technology
In Partial Fulfillment of the Requirements for the Degree of

MASTER OF SCIENCE

in Computer Engineering

by
Emrah İNAN

July 2012
İZMİR

We approve the thesis of **Emrah İNAN**

Examining Committee Members:

Prof. Dr. İsmail Sıtkı AYTAÇ
Department of Computer Engineering
Izmir Institute of Technology

Assist. Prof. Dr. Pars MUTAF
International Computer Institute
Ege University

Assist. Prof. Dr. Tolga AYAV
Department of Computer Engineering
Izmir Institute of Technology

03 July 2012

Prof. Dr. İsmail Sıtkı AYTAÇ
Supervisor, Deparment of Computer Engineering, Izmir Institute of Technology

---

Prof. Dr. İsmail Sıtkı AYTAÇ
Head of the Department of Computer Engineering

Prof. Dr. R. Tuğrul SENGER
Dean of the Graduate School of Engineering and Sciences

ACKNOWLEDGEMENTS

First of all I would like to express the deepest appreciation to my advisor, Prof. Dr. Sıtkı Aytaç, for his abundant help and encouragement. He continuously conveyed a spirit of adventure in regard to research and his inspiration, and great efforts during this research made this thesis valuable.

I also consider it an honor to work with Dilek Tapucu and Fatih Tekbacak in İzmir Institute of Technology Department of Computer Engineering. It was a pleasure to work in such elite colleagues and I would like to state my special thanks to their prolific suggestions.

In addition, I am indebted to thank my colleague Cemalettin Öztürk in Izmir University of Economics, for his support, motivation and enthusiasm.

I dedicate this thesis to my parents who support me during my life and my graduate study. They made this work possible.

ABSTRACT

MoresysGOAL: MOVIE RECOMMENDATION SYSTEM USING COLLABORATIVE FIILTERING TECHNIQUE SUPPLEMENTED BY CONTENT WITH GOAL PROGRAMMING

In recent years, internet grows at an accelerating rate. In addition, a new flow of information, which has various types of data, takes place at internet. Therefore, the end users may not find the relevant information satisfying their interests. As a result, recommendation systems, one of the approaches, appeared to help users for this manner. MoresysGOAL is one of the examples for these systems, and stands for movie recommendation system with goal programming. It aims to improve the state-of-art collaborative filtering algorithms unless they have enough dense dataset. Hence, MoresysGOAL has a successful combination of content-based and collaborative filtering approaches for increasing performance of the recommendation system.

This thesis focuses on serving a successful solution to users considering two parts. The first part is related to the similarity calculation of the contents are supplemented by goal programming. Moreover, the proposed system has the content information of the movies which also play a role to support collaborative filtering algorithms. These collaborative methods form the second part by means of predicting movies to satisfy user tastes. Lastly, MoresysGOAL is a web-based application for recommending prediction lists of movies to the end users.

ÖZET

MoresysGOAL: İÇERİĞİ AMAÇ PROGRAMLAMA İLE DESTEKLENMİŞ İŞ BİRLİĞİ TEKNİĞİNİ KULLANAN FİLM ÖNERİ SİSTEMİ

Son yıllarda internet hızlı bir şekilde büyümektedir. Buna ek olarak, çeşitli veri tipleri içeren yeni bir bilgi akışı internette yer almaktadır. Böylece, son kullanıcılar ilgilerini tatmin edebilecek bilgiyi bulmakta zorlanmaktadırlar. Sonuç olarak öneri sistemleri kullanıcılara bu zorluklara karşı yardımcı olmak için çıkan yaklaşımlardan biridir. MoresysGOAL öneri sistemlere bir örnektir, ve amaç programlama kullanan film öneri sistemi anlamına gelmektedir. Yeterince yoğun veriler içermeyen geleneksel iş birliğine dayalı algoritmaları geliştirmek amacını taşımaktadır. MoresysGOAL içerik tabanlı ve iş birliğine dayalı yaklaşımların başarılı bir kombinasyonuna sahiptir.

Bu tez iki kısımda düşünülerek kullanıcılarına başarılı bir sonuç sunmaya odaklanır. İlk durumda içeriklerin benzerlik hesaplamaları amaç programı yazılımı olan OPL ile bulunur. Aynı zamanda geliştirilen sistem iş birliğine dayalı algoritmaları destekleyen film içerik bilgisine sahiptir. Bu iş birliğine dayalı metotlar kullanıcıların isteklerine uygun filmleri tahmin ederek ikinci kısmı oluştururlar. Sonuç olarak MoresysGOAL film tahmin listelerini son kullanıcıya öneren web tabanlı bir uygulamadır.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Table **Page**

CHAPTER 1

INTRODUCTION

In today's world, internet grows at an accelerating rate and a new flow of information takes place at any time. Moreover, variety and type of this information change dramatically on internet. Therefore, users aim to find results matching their needs (Marmanis & Babenko, 2009). Actually users have to reach what they find out as fast as it can be. Because they must keep pace with innovations quickly changing order in the 21th century. Thus this kind of challenge has been regarded as a problem of information overload (Ghazanfar & Rügel-Bennett, 2010).

At the beginning of the last decade, internet has included several types of web sites which are basically concerned in e-commerce, interaction of people, actual news and information sharing (Su & Khoshgoftaar, 2009). According to this demand on internet, new terms such as information retrieval (Salton, 1989) and knowledge discovery (Fayyad et al., 1996) have existed. As a result of availability of huge data in the internet cloud, these terms provide an efficient but also hard way to extract the most valuable data to be transformed into an asset.

Internet, in other words, is a huge mountain then ready to be mining. Knowledge discovery serves some kinds of processes in which data mining (Han & Kamber, 2006) is a significant tool to reach the valuable knowledge. Information extraction (Banko et al., 2007) and information retrieval are other techniques.

Besides, when the information is obtained, it opens new doors for commercial web sites (Schafer et al., 2001). With the help of information extractor techniques and users interaction e-commerce web applications can recommend users what they really search in quite a short time. Therefore, e-commerce web sites can sell many more products than they expect in a day. As a result, they can find a chance to increase their profits and then these firms have a growth in the market. This growth also provides them to recommend new things for users to buy what they exactly want (Fleder & Hosanagar, 2007).

Specifically, based on the relationship between web applications in World Wide Web and users recommendation system term has appeared in the mid-1990's (Hill et al.,

1995). This term means that it is such a system that recommends items which are really interesting for the users (Alag, 2008). There are many web sites in today's internet world recommending various kinds of products. While (Amazon, 2012) shows the books to the clients, (Youtube, 2012) is a well-known web application for recommending videos; (MovieLens, 2012) and (IMDb, 2012) provide movies to user interest. In addition (Last.fm, 2012) is an example of music recommender system.

Due to the fact that this thesis mainly concerns in movies domain, Netflix (Bennett & Lanning, 2007) which is an online rental service for movie subscribers, has organized a competition for entrepreneurs in the United States, Canada, United Kingdom and Ireland. Netflix aims better movie predictions to the end user in terms of previous ratings given by them. Netflix has a huge dataset which contains more than 100 million ratings given by almost 480 thousand users to 18 thousand movies, whether a firm can improve the result more than ten percent, it will win the prize served by Netflix.

In order to recommend an item to a specific user, recommendation systems collect the information about user either implicitly or explicitly (Ziegler et al., 2005). Explicit process is performed by user awareness such as survey analysis, though implicit feedbacks are collected in background when users click the links or make comments about an item. Youtube asks the users to vote the videos and Amazon also stores the viewing times of the related item. The former one is an example of explicit feedback; the latter is concerned in implicit data collection.

From another point of view, recommender systems are also used as search engines because they provide similar items to users according to their needs. Some search engines can suggest limited queries and in this way recommendation systems may support them (O'Mahony et al., 2008). In other words, artificial intelligent systems and intelligent agents have a role in Web-based applications such as search engines and recommender systems (Russel & Norvig, 2010). At its core point of artificial intelligent systems, contains basically several exploration mechanisms and it gives machines a talent of thinking like a human mind.

Recommendation systems have not only concerned in the interaction of users and products, but also they have been an interdisciplinary methodology. They have a close relationship between natural and social sciences. Natural sciences provide the construction of mathematical background likewise approximation theory and recommendation system are also related to cognitive sciences, in which psychology is

one of the parts of this interdisciplinary scientific area (Adomavicius & Tuzhilin, 2005). By indirection, management science is another interesting example of social sciences that are related to recommendation systems (Murthi & Sarkar, 2003). Moreover, prediction techniques of the recommendation systems are studied in forecasting theories (Armstrong, 2001).

There are several drawbacks such as data sparsity and scalability in the subject of recommendation systems. In general, these problems are derived from low density of the dataset which includes large amount of data. Also, datasets used as an input for recommender applications diversify from different kind of information. In fact, our study aims to overcome these types of problems and try to recommend more reliable and correct movies to the end users.

1.1. Problem Definition

In general, large datasets are examined in many recommendation systems. The preference values given by users to specific items may not be enough in these datasets. In other words, low density of the datasets has appeared in several conditions such as new user or new item challenge (Yu et al., 2004). Furthermore, these problems might be called as cold start problem in other literature studies. In these kinds of situations, recommender systems cannot predict the preference of an actual user until some similar users of this actual one give a rating to new relevant items. As a result of insufficient past information for a new user, the system may not recommend accurate results.

This thesis mainly considers alleviating "sparsity" challenge of the datasets. In this situation, content information is used to predict preference values for new users who have interested in new items. Moreover, this content-boosted method is supplemented by hybrid techniques.

1.2. Purpose of Thesis

Initially, *MoresysGOAL* stands for movie recommendation system with goal programming. It aims to improve the state-of-art collaborative filtering algorithms unless they have enough dense dataset. Hence, *MoresysGOAL* is kept contact with a successful combination of content-boosted collaborative filtering application for movie

recommendation (Özbal et al., 2011) and effective missing data prediction (Ma et al., 2007) for increasing performance of the recommendation system. Because both of these approaches have an important achievement for overcome the drawback of data sparsity and also scalability problems.

This thesis mainly focuses on serving a successful solution in which contents are supplemented by Optimization Programming Language (OPL). It is a web-based application for recommending movies to the end users. Moreover, our system contains content information of the movies which also play an important role to support collaborative filtering algorithms by means of prediction better movies to satisfy user tastes.

1.3. Structure of Thesis

Structure of *MoresysGOAL* is stated as follows; Chapter 2 is related to the recommendation systems in a more detailed way. It introduces a comprehensive exploration of this kind of systems dealing with popular recommender systems used in daily life are discussed in detail.

In Chapter 3, four main approaches, collaborative filtering, content-based, demographic and knowledge-based used in recommendation systems are presented in terms of advantages and disadvantages. Further hybrid recommendation systems which focus on different combinations of these four techniques are covered.

Chapter 4 includes theoretical background of *MoresysGOAL* consisting of prediction technique behind the recommendation systems which focuses on a combination of collaborative filtering and content-based approaches.

Chapter 5 handles the whole system design and software architecture used in *MoresysGOAL*. There are three applications; two of them are preprocessing applications that gather contents and one main web-based application to provide recommended lists for the users.

Chapter 6 evaluates the performance of *MoresysGOAL*. At the beginning of this chapter, domain model is presented and then experimental process is run for measuring achievement of *MoresysGOAL*. Besides, the results are comprehensively compared with other successful applications in the literature.

Finally, Chapter 7 reveals a summary of *MoresysGOAL*, *i.e.*, it has not only concluded overall background technology and algorithms worked in *MoresysGOAL*, but also has given possible contributions and improvements in the future.

CHAPTER 2

RECOMMENDATION SYSTEMS

Initially, recommendation system is a kind of information filtering system suggesting relevant items to the users with the help of intelligent agent systems. In addition, suggested items have several options such as movies, books and news (Adomavicius & Tuzhilin, 2005). In this situation, item and product are abstract terms commonly used to define inputs of recommendation systems (Ricci et al., 2011). More formal definition of recommendation system (Mahmood & Ricci, 2009) is *"recommender systems acquire the users' preferences, and use them to build some type of a user model. Then, the system predicts the set of products that best "matches" the user model, and recommends them to the user".* According to the definition, user model is a term used for giving preference values to a specific user. Each specific user can increase the quality of the recommendation system when collecting any information satisfies user's tastes.

Later, with the help of user modeling, the system predicts preferences or ratings for the items. In other words, recommended items can be either product such as a movie CD or social element like a group.

In a formal way, it can suppose that a set of users $U = \{u_1, u_2, \dots, u_m\}$ and a list of items $I = \{i_1, i_2, \dots, i_n\}$. Then user profile $u \in U$ is seen as *n* dimensional vector that consists of ordered pairs (Mobasher, 2007)

$$u^{(n)} = \langle (i_1, s_u(i_1)), (i_2, s_u(i_2)), \dots, (i_n, s_u(i_n)) \rangle \quad (2.1)$$

Where for user u, s_u is rating function, $i_j's \in I$ given utility values to items in I are also for this user. In general, recommendation system is made up of mxn matrix $UP = [s_{u_k}(i_j)]_{mxn}$ in here $s_{u_k}(i_j)$ shows the level of user u_k interest on item i_j. Hence, recommender engine indicates a mapping $REC : P(UP)x\ U \rightarrow P(I)$ according to the profiles of users a list of items are mapped by each user. In an assumption, for an actual user $u_k \in U$ item that has the best predicted rating is provided and more formally a general formulation of a recommendation system is stated as below

$$REC(up, u_k) = \left\{ argmax_{i_j \, \epsilon \, I} \, s_{u_k}(i_j) \right\} \tag{2.2}$$

Where, up shows the subset of user profiles set (UP), and $s_{u_k}(i,j)$ is the preference value of item i_j that is predicted by the user u_k. Characteristically, these kinds of systems recommend top N list items sorted by preference values. According to the question which prediction technique is used, the recommendation engine might not give a preference value for a specific item, in this condition REC mapping may not produce a a number (*NaN*) or *null* preference value.

Table 1 indicates that five users Brian, Steve, Sarah, Bill and Jessica gave ratings to four movies Scarface, The God Father, Pulp Fiction and The Shawshank Redemption.

Table 1. Basic User-Movie Matrix

	Scarface	The God Father	Pulp Fiction	The Shawshank Redemption
Brian	3	0	1	4
Steve	2	5	0	0
Sarah	4	3	0	2
Bill	0	2	4	1
Jessica	0	4	3	2

Instead of null preference values, zero is assigned for previously unmonitored movie. For example, Brian has not seen The God Father yet and then this null value is set to 0. As seen in Table 1, preference values of users have a rating scale from 1 to 5.

At the beginning phase of recommendation systems, it aims to find the most convenient rating for this 0 valued actual rating. The most appropriate way of considering this situation is to predict new rating values to the movies that have not seen by the end users yet. Next step of recommendation system is suggesting more accomplished recommendations to the users. Prediction algorithm can provide a predicted list of the movies in top down order. This is also called top-N lists in descending order served for user tastes (Zhang, 2009).

Generally inputs, recommendation construction and outputs are then three main parts of these kinds of systems. Items are the objects of recommendation systems and

they are served to the end users. They may have positive or negative meanings for an actual user. Indeed, recommender engine must understand the most appropriate item for that user. Users are the subjects and they may have several characteristics. Furthermore, based on the type of recommendation system they might be objects.

The user information can be collected in many situations such as behavior pattern, demographic and social networking. Ratings or votes are popular transaction datasets between users and recommender systems.

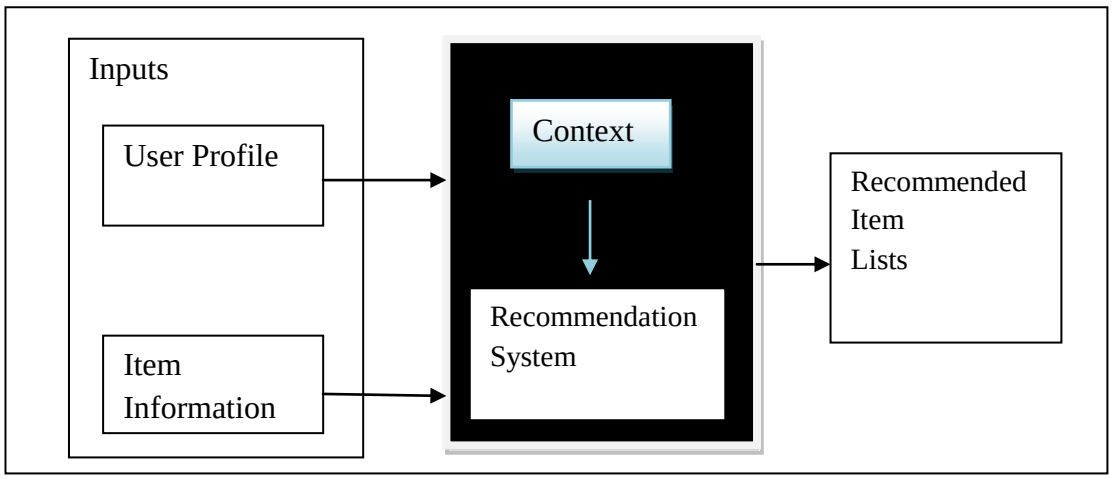

Figure 1. Inputs and Outputs of a Recommendation System

Figure 1 gives high level information about a recommendation engine (Alag, 2008). Inputs can be collected in two ways; these are users and items. A user can be considered as an item and vice versa. In collaborative filtering or social networking approaches relevant people might be recommended to the current user based on the other users' profiles. The user profile consists of demographic information about actual user such as age, gender, location, etc. Users' contents vary from ratings to tagging. They especially send e-mail to each others, bookmark a relevant web site or browse content on the internet.

The users' interactions feed the collaborative filtering techniques. Further, contents are related to the items that have a role as an input in these kinds of techniques. In addition, *"Top Item List"* term is used for creating personalized list such easy form where products that have been purchased, pinned or viewed recently are served for satisfying the actual user's taste.

At this moment, a confliction occurs between exploitation and exploration in recommendation systems. Exploitation is the process of recommending products based on items already known by the current user. Another term is exploration in which recommender engine aims to present new items that current user might like. Here there exists an infinite loop because these new items will be also used for exploration in near future. Therefore, a solution in which some randomly selected new items can be added to the top item list may overcome the exploitation problem.

2.1. Recommendation System Methodology

General methods of the recommendation systems focus on the prediction processes examined under the condition of selection of item or user perspectives. If these methods are only related to the content features of the items, they are called content-based or information-based systems. In other cases, prediction process may be manipulated in social background of the users. In the last point of view, combination of both prediction processes can also be handled. Therefore, there are four main parts of the recommendation systems methodology (Burke, 2007)

- Content-based
    - Content information of the items and the preference values given by users are both domains of content-based recommendation systems.
- Collaborative
    - It concerns neighborhood of user pairs; then this neighborhood similarities are used for recommending relevant.
- Demographic
    - Demographic information that is the basic component when providing recommendations includes gender, age, occupation, location information for a specific user.
- Knowledge-based
    - It considers needs and preferences of users and according to those inferences it suggests recommended items.

In addition, a combination of these types of recommendation systems entails the term hybrid recommender system and in the following sections hybrid systems are analyzed in detail.

2.2. Content-Based Recommendation Systems

Content-based recommendation systems tend to investigate descriptions of the items which are preferred by the particular users. If there is a similarity among the observed items, actual users may probably give equal preference values for those items which are related to these test users. Before the beginning of the similarity calculation process, item representation has a role to determine the details of recommendation systems (Pazzani & Billsus, 2007).

Table 2. Simple Database Content Table Example

ID	Name	Cuisine	Service	Cost
10001	Mike's Pizza	Italian	Counter	Low
10002	Chris's Café	French	Table	Medium

Table 2 indicates general information about the structure of a simple database table that has two records or rows explaining two restaurants. There are 5 columns in the database table which are also called "attributes", "fields", or "variables" in the other literature studies. Moreover, ID attribute has a unique value for each record in the database table. This example of the structured data may include few properties and may be then used for recommending restaurants to the users. With the help of structured data, this kind of limited description may provide creation of a user profile easily. Another concept for content-based recommendation systems is user profiles that include interests of the users, are concerned in two situations which are (Pazzani & Billsus, 2007)

- The preferences of users' model contain explanatory representations of kinds of products that satisfy those users' interests. Many possible alternative notations may occur for this explanation. However, a well-known method in this situation is to find similarity of the user who has an attention in that product.

- In this situation, it concerns the past interactions of the users observed in the recommendation system. This may diversify in types such as purchased product by the user or given rating to the items.

In content-based recommendation systems, historical dataset is also used for a user model that is training dataset of a machine learning technique. Moreover, user customization and rule-based recommendation engines which are other important terms that are related to this topic are discussed in the following sections.

2.2.1. Disadvantages of Content-Based Recommendation Systems

Content-based recommender systems provide items which have shallow content information and for obtaining enough features, system has to automatically parse such content as text files. Another long way is to assign features of products manually (Balabanovic & Shoham, 1997). Further, some kinds of datasets such as multimedia data, audio and video streams are hardly practiced by automatic feature extraction techniques. Also, restrictions of domains assigned features by hand may not easily applicable (Ahn & Shi, 2008).

Another weakness is *"indistinguishable"* content problem that occurs when different item pairs are represented by the same kind of attributes. For instance, it cannot be clearly found out defining difference between well-written and badly written articles whether both of them are represented by similar keywords that have same importance for each article in this pair (Ahn & Shi, 2008).

In the occurrence of overspecialization problem, the profile of an actual user is constructed by the past interactions of this current user with other users or items. Due to this user's profile, recommendation system can only give recommendations taken high values. But the system ignores *"randomness"* situation because it gives probably recommended items which have a high similarity already seen items. Genetic algorithms are examined for providing random alternatives for the actual user to overcome this weakness (Sheth & Maes, 1993).

At the beginning of prediction phase of content-based recommendation system, domain of this system has to include satisfactory preferences of users for sufficient

products. Hence, this system has a little chance to increase accuracy of recommended items (Sheth & Maes, 1993).

A collection of five redundancy measurements is served by the study (Zhang et al., 2002) to examine relation of documents containing some information novelty. Lastly, diversity is also an element for content-based recommendation systems. Heterogeneous alternative and *"range"* of options should be provided by a successful content-based recommendation system.

2.2.2. Advantages of Content-Based Recommendation Systems

Content-based recommendation systems have little strength (Ahn & Shi, 2008). For instance, this type of recommender system can be easily understood and it is mostly related to the user feedback. Therefore, recommendation process of this kind of system has a high reliability.

Computation process of these recommender systems consumes small piece of resources and so content-based recommendation systems are cheap systems. Moreover, content-based approaches are lightweight systems because they cannot need personalized service that is without content of user interest. Finally, these systems are highly coverage, in other words, they have wide range of items for recommendation process.

2.3. Collaborative Filtering Recommendation Systems

Collaborative Filtering (CF) recommendation systems investigate the similar users of an actual user rather than searching similar items that user has liked before. The term "nearest neighbor" occurs exactly in this situation. This term concerns in finding high value of correlation when looking forward the previous preferences of those users. Experiences of an actual user's friends have a good indicator for common preferences of this user. In addition, *"Word-of-mouth"* term exists when Collaborative Filtering techniques make recommendation processes automatically (Shardanand & Maes, 1995).

Collaborative Filtering methods have the following basic assumption; whether *X* and *Y* users give similar ratings to *n* items or indicate common behaviors such as

watching movies or listening to music and as a result of this recommender system may give similar preference values to these users for other products (Goldberg et al., 2001).

Most of the Collaborative Filtering techniques have an array of *m* users $\{u_1, u_2, \dots, u_m\}$ and *n* items' list $\{i_1, i_2, \dots, i_n\}$ and there is also a list of items for each user u_i ,I_{u_i} that user has already given ratings (Su & Khoshgoftaar, 2009).

There are two types of gathering user interest processes which are explicitly and implicitly collecting preferences of users. In the explicit data collection way users can give a vote in a range between 1 and 5. This process is completely controlled by user pleasures. The second one is indirection way of data collection clique-based links or purchased items are examples of implicit indications (Miller et al., 2004). General algorithm of Collaborative filtering can be stated as below (Ahn & Shi, 2008)

1. *Extraction of rating profiles of users are used for evaluating already experienced items.*
2. *By examining this user rating profiles similarity measurements of those user pairs can be calculated by an efficient algorithm.*
3. *Based on this similarity measurements current user's preference value is predicted for a new item.*
4. *A predicted list of preferences provided by a descending order is recommended to current user.*

Collaborative Filtering techniques such as memory-based and model-based are explained in the following sections in detail.

2.3.1. Memory-Based Collaborative Filtering Methods

In the prediction process of memory-based Collaborative Filtering methods, the whole or a part of user-item ratings matrix is used as seen in Table 3 below.

Table 3. Rating Values of User-Movie Matrix

User-Item ratings	$Item_1$	$Item_2$
$User_1$	3	5
$User_2$	4	2

According to calculation similarities of users the system constructs groups of people for each user. Then, with the help of those neighborhoods recommender engine predicts new items for this actual user. Actually, memory-based approaches that are also called neighborhood-based Collaborative Filtering methods can be summarized as below (Sarwar et al., 2001)

1. *Calculation of weights, $w_{i,j}$, that is implication of correlation between two users, i and j.*
2. *Predicted preference value is then served to the actual user by calculating account average of weights of the whole preferences of users for specific item.*
3. *If the system produces top-N list recommendations k nearest neighbors of this actual user have to find and then relevant calculation of similarity values and neighbors gives the top-N recommended item list to the new user.*

These three steps consider user perspective of memory-based collaborative filtering approaches. There is also another condition that all of these steps are used to make predictions based on item neighborhoods.

2.3.2. Model-Based Collaborative Filtering Methods

Model-based Collaborative Filtering approaches provide a model which is constructed by training process of ratings gathered by the user item interactions. Then, the model-based system uses this model to predict ratings for new unseen products to the end users. The main advantage of these types of systems is that they can overcome the scalability problem. However, they consume much more time when building or updating processes are made for the model.

Aspect model (Hofmann & Puzicha, 1999) is one of the pioneers of model-based approaches. A probabilistic latent-space model is obtained for this model. The User Rating Profile is another example of the models that has a latent factor model which mixes preferences of users and then these mixing levels may be different based on a Dirichlet random variable (Marlin, 2003).

In addition, clustering methods (Ungar & Foster, 1998) are used for defining user groups who have similar attitudes. Once the model of clusters is created and then

recommendation system produces predicted values according to the average of the clusters whose weights are calculated in level of association.

2.3.3. Strengths and Weaknesses of Collaborative Filtering Methods

In conclusion, Table 4 summarizes the main collaborative filtering methods and it gives also main strengths and weaknesses (Su & Khoshgoftaar, 2009)

Table 4. Collaborative Filtering Algorithms
(Source: Su & Khoshgoftaar, 2009)

CF Categories	Representative techniques	Main Advantages	Main Shortcomings
Memory-based CF	*Neighbor-based CF (item-based/ user-based CF algorithms with Pearson/vector cosine correlation) *Item-based/user-based top N recommendations	*easy implementation *new data can be added easily and incrementally *need not consider the content of the items being recommended *scale well with co-rated items	*are dependent on human ratings *performance decrease when data are sparse *cannot recommend for new users and items *have limited scalability for large datasets
Model-based CF	*Bayesian belief nets CF *Clustering CF *MDP-based CF *latent semantic CF *Sparse factor analysis *CF using dimensionality reduction techniques, for example, SVD, PCA	*better address the sparsity, scalability and other problems *improve prediction performance *give an intuitive rationale for recommendations	*expensive model-building *have trade-off between prediction performance and scalability *lose useful information for dimensionality reduction techniques

(cont. on next page)

Table 4. (cont.)

Hybrid recommenders	*Content-based CF recommender, for example, Fab *content-boosted CF *hybrid CF combining memory-based and model-based CF algorithms, for example, Personality Diagnosis	*overcome limitations of CF and content-based or other recommenders *improve prediction performance *overcome CF problems such as sparsity and gray sheep	*have increased complexity and expense for implementation *need external information that usually not available

2.4. Demographic-Based Recommendation Systems

Demographic features such as gender, age and salary, etc. for users are materials to learn basic human behavior. Gathering of personal information is a specific process in which many of them naturally do not want to share their private information for the public online web sites. If these kinds of data can be collected in a reliable way, they can be useful for preparing recommendation systems. Moreover, Table 5 indicates a generic table of demographic-based system;

Table 5. Sample Demographic Data Table

	Age	Gender	Location	Occupation
Bill	25	M	USA	Lawyer
Sarah	28	F	UK	Engineer
Jessica	16	F	Germany	Student

In Table 5, it is obviously seen that these private contents are mostly used in recommender systems in e-commerce applications. Demographic contents for user *u* are used to find similarities between other users to recommend an item in list *I* based on the calculated similarities.

Most of the applications are combination of demographic content and collaborative filtering approaches. Demographic generalization (Krulwich, 1997) is employed to obtain users profiling and discover the general preferences of users based on these classifications. Therefore, it is obviously revealed that demographic information is a supplementary material for other important techniques.

2.5. Knowledge-Based Recommendation Systems

Knowledge-based recommender systems provide items according to the preferences of the users. In general, these kinds of systems deal with valuable knowledge of users and with the help of these data the system tries to forecast the most interesting products for an actual user (Burke, 2000).

Content information of products is stored in list *I*. Then these features are used for satisfying user tastes. Inputs are naturally explanation of interests of user *u*. Then output provides the best matching between user *u* and item *i*.

Knowledge-based recommender system is also a part of hybrid approaches like demographic filtering in general. They use a supplementary role combining with content-based or collaborative filtering techniques because of their advantages. One of the main properties is that they can reduce the problem of prediction item for new user entrance. Reason of this strength is that these kinds of systems do not require the ratings of users. Knowledge-based methods draw a general figure about users as are result they consider independent conditions from specific user interests.

2.6. Hybrid Recommendation Systems

Hybrid recommendation system is used for combining features of different types of recommendation systems to aim performing better output results. In the previous sections, the advantages of several recommendation systems were explored and based on these advantages, seven different types of hybrid recommender systems are introduced (Burke, 2007)

- *Weighted:*
    - *It is most probably the simplest structure for a hybrid recommendation engine. Each method of hybrid system gives output for a current item and then these outputs are used for a linear formula.*

- *Switching:*
    - *According to the current recommendation condition, switching hybrid method selects a single recommendation among its elements.*

- *Mixed:*
    - *It provides recommended lists alongside combined by its different elements.*

- *Feature Combination:*
    - *It is a single recommendation system derived from features taken from many domains.*

- *Feature Augmentation:*
    - *It is a sequential application in which the results of the first type of recommendation system used as inputs of the second one.*

- *Cascade:*
    - *The results taken from component of strictly hierarchical hybrid is improved by other recommender algorithm.*

- *Meta-level:*
    - *In this concept, one recommendation engine produces a model and then this model is used as input for another recommender.*

There are many conditions that different methods of the same kind may be used for hybridization (Basu et al., 1998). For instance, two different techniques of content-based recommender algorithms can be used for *switching* method. In the seven different types, combinations mainly focus on taking properties of the techniques that aim to overcome problems such as new user or new item entries. There is also a review of these combinations of recommendation systems as seen in Table 6 (Burke, 2000).

Table 6. The Space of Possible Hybrid Recommender Systems
(Source: Burke, 2000)

	Weight	Mixed	Switch	FC	Cascade	FA	Meta
CF/CN							
CF/DM							
CF/KB							
CN/CF							
CN/DM							
CN/KB							
DM/CF							
DM/CN							
DM/KB							
KB/CF							
KB/CN							
KB/DM							

Where FC stands for Feature Combination and FA means Augmentation in the columns. The abbreviations in the rows are; CF is collaborative filtering, CN is content-based, DM is demographic and lastly KB is knowledge-based one.

	Not Existing implementation
	Existing implementation
	Redundant
	Not possible

Table 6 shows 53 possible two-part hybrid recommendation systems. In this table, existing and not existing implementations are taken into account by the researchers. In these conditions, there are many applications that will be introduced in the following chapter.

2.7. Comparison of Hybrids with Single Recommendation Systems

With the help of using different kinds of domains such as content, collaborative and demographic datasets, the study (Basu et al., 1998) is important for comparison of weighted and meta-level hybrids. Another study (Melville et al., 2002) is a combination

of content-based, collaborative and knowledge-based methods for recommending movies. This study clearly indicates that hybrid recommendation systems are better than many methods.

Before comparing hybrid recommendations with four categories of recommendation systems those are content-based, collaborative, demographic and knowledge-based; pros and cons of these types of systems have to be identified. Content-based and collaborative filtering approaches can easily use if there occur enough preference and item dataset.

CHAPTER 3

RELATED WORK

This chapter focuses on the related work in the area of the recommendation systems. Several applications will be reviewed from different types of resources in four main categories; these are content-based, collaborative filtering, demographic-based and knowledge-based approaches. Moreover, hybrid recommender systems which are combination of previous four recommendation system types will be discussed in detail. This chapter also offers finding solutions to common problems in recommender system when advantages and disadvantages of these techniques are discussed.

3.1. Content-Based Recommendation Systems

Pure content-based recommendation systems regard the prediction phase as a categorization of text problem to serve content-based predictions. Movie content information is indicated like a document and ratings given by users from 0 to 5 construct six labels of class (Mitchell, 1997).

A bag-of-words naïve Bayesian text classifier (Good et al., 1999) can be handled and each movie feature such as title, director, etc. is equivalent to a bag-of-words. Then this classifier is used for learning a profile of user from labeled documents. Afterwards this user profile has been entered a process that is the prediction of rating of watched the new movie. In a similar way, LIBRA (Debnath et al., 2008) has been used successfully when recommending new books. Instead of using 0-5 rating scale, ratings used in LIBRA have been distributed between 1 and 10. The ratings which are greater than 5 mean positive. In addition, the ratings less than or equal to 5 have negative meaning. According to the positive probability calculation, similar to the previous application user profile is obtained from labeled documents. Then, this profile is used for ranking all other books as recommendations.

RIPPER (Goldberg et al., 1992) stands for repeated incremental pruning to produce error reduction, which is an example of rule induction algorithm related to the

decision trees. The system in these decision trees has similar principles with the recursive data partitioning.

The content-based algorithms use information about an item to suggest recommendations to the users. With the help of social networking and collaborative filtering approaches, a classification method is proposed as a recommendation system (Good et al., 1999). This system is an inductive learning approach to provide recommendations to the end users. It is capable of using ratings and other forms of information about the preference of each user.

Experimental results of the proposed approach (Good et al., 1999) show that this application performs better results than traditional approaches. This system provides flexibility used for exploitation by using content included two types of representations. The effect of multiple information sources gives more accurate recommendations rather than exploiting a narrower amount of content.

In the feature weighting in content-based recommendation system (Debnath et al., 2008), weights are assigned for these features due to the importance for the actual users. Then, a linear regression formula is handled by a social network graph which gives information about the judgments of person for calculating item similarity. This system also uses a combination of content-based and collaborative filtering methods.

Debnath *et al.* (2008) suggests that any preference data does not need. It serves recommendations according to the item similarities calculated by using suitable distance measures of item attributes. As a result, similarity measure of item is then used in content-based algorithm which is supplemented by a collaborative social network of the end users.

Feature weighting system (Debnath et al., 2008) selects item attributes from IMDb database for movie recommendations such as genre, cast, writer, etc. Due to the interest of users for these features, several weight values are set by using regression analysis. Therefore, this regression equation is used for computing similarity of items. It is seen in the empirical analysis; the proposed method gives better results than other state-of-art content-based recommendation systems.

3.2. Collaborative Filtering Recommendation Systems

Collaborative filtering phrase has publicized as discussed in the previous chapter in detail. One of the first recommendation systems *Tapestry* (Goldberg et al., 1992) is an experimental mail system recommends messages to the actual user when selecting relevant document in terms of collaborating this test user with other users.

Another pioneering and ongoing effort is GroupLens that uses user ratings for the weight computations of users or items and then predicts ratings of new items based on those weights (Resnick et al., 1994). This application is also one of the first memory-based recommendation systems explained in section 3.2.1. Based on the similar news reader clients for an active user, the recommendation engine presents articles to the actual user.

A pure collaborative filtering approach can use an algorithm based on neighborhood (Herlocker et al., 1999). This kind of application selects the most similar users for a test user. Then weights of their ratings are calculated and lastly, combination of them is used for prediction process to the test user.

There are examples applied in different domains. For instance, Ringo (Shardanand et al., 1995) considers the music recommendations and Jester (Goldberg et al., 2001) uses really interesting resource, is related to the jokes. Furthermore, (MovieLens, 2012) is web-based application that serves movie recommendations to the users. Next two sub sections give detailed examples of both memory-based and model-based collaborative filtering approaches.

3.2.1. Memory-Based Collaborative Filtering Methods

The oldest examples of collaborative filtering approaches are related to the memory-based systems. Memory-based applications should be studied in two ways these are user and item. Initially several item-based approaches are introduced then user-based applications are discussed in detail.

Amazon.com (Linden et al., 2003), item recommendation system is an example for e-commerce web applications. It focuses on matching similar users to an actual user. Amazon first explores similarity between other items and each purchased or rated products of this user, afterwards it gives a list of recommendations based on the

combination of those similar items. Linden *et al.* (2003) claim *"Customers who bought items in your Shopping Cart also bought"* technique of Amazon.com gives an inspiration for e-commerce web sites nowadays.

One of the main strengths of item-based collaborative filtering approaches is that it decreases large amount of dataset when computing similarity weights. For instance, *MovieLens* dataset has approximately 7-fold more users than items. In this case, item-based collaborative filtering can naturally decrease the scalability is a well-known problem in collaborative filtering techniques.

(Last.fm, 2012) is another web-based application and aims to recommend tracks to the end users. Each user can listen or scrobble (The Scrobbler sends little note about which song the listener is playing at this moment) a track recorded by a musician. This system has collected data in two ways; these are music content and user profile. Hence, both user and item-based similarity values can easily be calculated. In addition, Last.fm gives a chance to the developers who want to build applications with the help of freely available (Last.fm API, 2012).

Lastly (Grooveshark, 2012) is an example of accomplished music recommendation system and is the largest music discovery service in the world. In the databases of Grooveshark, it has more than 15 million songs and 35 million users across the cloud.

The key step of many memory-based algorithms is to find the similarity of user pairs. In this condition, both local and global user similarity approach provides an application of the term of *maximin* distance in graph theory according to the *surprisal-based vector similarity* (Luo et al., 2008). Here, surprisal mentions the information quantities related to local and global users' ratings. As a result of this, surprisal-based vector similarity defines the relationship of user pairs.

At this point, whether user pair is connected through their locally similar neighbors, global user similarity considers that this user pair can be similar. Luo *et al.* (2008) propose one of the collaborative filtering methods that outperform other state-of-the-art approaches. Under the condition of sparsity problem, there are not enough similar neighbors and ratings for a specific product. Global user similarity is addressed for this problem. It is seen that if there are few ratings of the test users or few training users, the global user similarity will also contribute to increasing the accuracy of rating prediction.

Effective missing data prediction (EMDP) (Ma et al., 2007) application is an example of memory-based collaborative filtering technique. This approach mainly relies on Pearson correlation coefficient (PCC) in such a way that a new parameter is inserted to protect the system form inaccurate similarity calculations of users or items. The second factor of this algorithm is the effective missing data process. This process considers the users and items together. Moreover, similarity threshold values for users and items. Hence the system can decide if a missing data is predicted or not.

A combination of contents about user and item are both used for prediction process of the missing data. In the experimental setup, it is obviously revealed that effective missing data prediction algorithm outperforms sparsity problem by increasing diversity in a more accurate way.

3.2.2. Model-Based Collaborative Filtering Methods

Eigentaste (Goldberg et al., 2003) uses a successful dimensionality reduction method. The main purpose of *Eigentaste* is the sparsity problem in terms of examining universal queries rather than using user-selected ones. The difference is that universal queries have a short un-biased description such as book summary, film synopsis, etc. Therefore, the system can able to provide density resulting subset of ratings matrix.

Cluster-based smoothing algorithm is a combination of model and memory-based approaches (Xue et al., 2005). Rating dataset of a cluster of similar users for unrated items can be predicted for a specific user in this cluster. As a result of this, missing data can effectively be predicted. In addition, the nearest neighbor of the actual user might be in Top N most similar clusters and then the nearest neighbors can be selected in the Top N clusters. Therefore, this kind of collaborative filtering approach overcomes the scalability problem.

The system serves smoothing procedures to find a solution for the missing data prediction by using clusters (Xue et al., 2005). With the help of missing data prediction process cluster-based smoothing method solves the sparsity problem. As a result of this, this approach also provides more accurate recommendations to the end users. To summarize, cluster-based smoothing application gives better empirical analysis results than traditional collaborative filtering approaches. A generative probabilistic framework which behaves each user item ratings as predictors of unrated items is presented with

advantages (Wang et al., 2006). This approach concerns in both user-based and item-based collaborative filtering algorithms by combining them with similarity fusion.

The last predicted rating comes from three sources of fusing predictions (Wang et al., 2006); one of them is due to the ratings of the other users for the same item, the second is due to the different ratings given by the same user and the last one is predictions by other but similar users for other but similar items.

The different rating types are taken into consideration when similar users' additional ratings for the similar items are used for smoothing the predictions. As a result, the whole model overcomes the sparsity problem. Indeed, experimental results indicate that this probabilistic framework outperforms the traditional collaborative filtering algorithms (Wang et al., 2006).

Singular value decomposition (SVD) is a good example of matrix decomposition approach and it is contributed by a Bayesian approach to reduce overfitting in SVD where priorities are defined and with the help of variations of inference, all parameters are integrated out (Lim & Teh, 2007). This approach is successfully implemented and consumes less time in runtime and requires less memory storage.

3.3. Demographic-Based Recommendation Systems

In terms of privacy sensitivity to the demographic resources are not enough to be evaluated in recommendation systems. Therefore, various collaborative filtering or content domains are needed to boost demographic-based recommender engines.

LIFESTYLE FINDER (Krulwich, 1997) is one of the pioneering instances in these kinds of approaches. By leveraging a large amount of demographic data, this approach achieves efficient usage of a narrow user dataset when profiling users. Then, the main advantage of this application is that it can easily protect cold start problem. Although this technique overcomes the sparsity problem, it decreases accuracy of the overall predictions.

3.4. Knowledge-Based Recommendation Systems

It is obvious that pure knowledge-based applications might not be found easily. They are either assisted by collaborative or content-based filtering approaches. One of the pioneers of this approach is (PersonalLogic, 2012) system aims to support users when they decide to products such as car, computers and pets, etc. Further, this system tries to help users for their career opportunities. *PersonalLogic* collects the requirements of the buyers for a specific product and with the help of knowledge-based algorithm; it aims to find relevant products satisfying user tastes.

The main advantage of these type systems is that they do not require user rating dataset. These kinds of recommenders give strength in other perspective that they can easily adapt to the drastic changes in user behaviors due to the time because these systems do not need to collect information of previous habits of the users. However, important disadvantage of knowledge-based methods is that they must explore the crucial features of the products in detail.

3.5. Hybrid Recommendation Systems

There are several hybrid applications such as *MOVIES2GO* (Mukherjee et al., 2001) in which combines ratings of the items with the movie feature and uses semantic and web content. *Hydra* (Spiegel et al., 2009) is a web-based movie recommendation system, is a hybridization of collaborative and content-based filtering approaches. MovieLens user item rating dataset is used in Hydra database and it is also supplemented by content features of movies served in IMDb.

In addition, *Hydra* focuses on reducing system runtime cost in terms of supporting by singular value decomposition (SVD) algorithm. SVD is one of the methods passed in the dimensionality reduction in recommendation systems. The main aim is to factorize matrix and to obtain narrower effective dataset as an input for prediction process. Further, demographic information of users are considered when retrieving data from MovieLens. It is clear that privacy policies are paid attention by MovieLens researchers while they have collected data such as age, gender, etc. for relevant users.

Fab (Balabanovic & Shoham, 1997) is one of the precursor applications that combine social-filtering and content-based approaches. It is an agent-based application and it gathers data for preferences entered by users. User agents deal with each user and in other way collection agents are related to the documents. Therefore, new profiles are created by these agents and then new profiles are consolidated by the user's interactions.

Unified Boltzmann machines (Gunawardana & Meek, 2009) are probabilistic models and aim to produce more robust results with the help of hybridization of both collaborative and content-based approaches. These features are used for learning weights which are then performed to predict user actions. This system is accomplished by recommending cold-start items. Empirical analysis shows that unified Boltzmann machines outperform traditional collaborative techniques.

A hybrid recommender system for dynamic web users, combines collaborative and content-based approaches and processes of this system is handled in both offline and online phases (Nadi & Bagheri, 2011). This application uses combination of fuzzy cluster mean (FCM) and ant based clustering algorithms and with the help of offline processes the system serves appropriate recommendations to the test users online. The system first analyzes preferences of users in terms of taking advantages of content-based and collaborative filtering methods.

Content-Boosted collaborative filtering approach (Melville et al., 2002) aims to improve recommendations with the help of strengths of both content-based and collaborative filtering algorithms. The former is used for prediction phase of contribution with the existing user data, the latter is then suggests more personal solutions through collaborative filtering approach. This application gathers data from (EachMovie, 2012) dataset that is old version of MoviLens is used for collaborative filtering processes and IMDb dataset is studied as input for content-based phase.

The main purpose of Content-Boosted collaborative filtering approach is divided by two processes; the first one is converting a sparse user ratings matrix into a full ratings matrix with help of content-based predictors and the second one is providing recommendations by using collaborative filtering approaches. The proposed method outperforms other pure content-based and pure collaborative filtering techniques based on the empirical results (Melville et al., 2002).

Another hybrid method is the content-boosted collaborative filtering approach also used for movie recommendations (Özbal et al., 2011).Further, this system is

contributed by missing data prediction and local and global similarity. The proposed method mainly concerns in the data sparsity problem. With the help of combination of accomplished approaches of local and global similarity and missing data prediction, Ozbal *et al.* offers a solution for the data sparsity problem.

REMOVENDER (Özbal et al., 2011) is an example of web-based recommendation system for providing movies to satisfy users taste. It collects several features of movies by IMDb database for using to calculate item similarity. The current movie's language, country, cast or writer features have different importance for the end users. Then, a regression equation is obtained in terms of computation of the feature weights. Furthermore, content similarity calculation contributes the item-based collaborative algorithm effectively in the missing data prediction process.

In the experimental setup, it is obviously revealed that the proposed method outperforms the traditional methods and also effective missing data and local and global similarity approaches. Table 7 gives a detailed summary of the recommender system research area (Adomavicius & Tuzhilin, 2005)

Table 7. The Summary of Recommender Systems
(Source: Adomavicius & Tuzhilin, 2005)

Recommendation Approach	**Recommendation Technique**	
	Heuristic-based	Model-based
Content-based	Commonly used techniques: • TF-IDF(information retrieval) • Clustering Representative research examples: • Lang 1995 • Balabanovic & Shoham 1997 • Pazzani & Billsus	Commonly used techniques: • Bayesian classifiers • Clustering • Decision trees Artificial neural networks (ANN) Representative research examples: • Pazzani & Billsus 1997 • Mooney et al. 1998 • Mooney & Roy 1999 • Billsus & Pazzani 1999, 2000 • Zhang et al. 2002
Collaborative Filtering	Commonly used techniques: • Nearest Neighbor (cosine, correlation) Representative research examples: • Resnick et al. 1994 • Shardanand & Maes 1995	Commonly used techniques: • Bayesian networks • Clustering • Artificial neural networks • Linear Regression • Probabilistic models

(cont. on next page)

Table 7. (cont.)

Hybrid	Combining content-based and collaborative components using; Linear combination of predicted ratings Various voting schemes Incorporating one component as a part of the heuristic for the other Representative research examples: • Balabanovic & Shoham 1997 • Claypool et al. 1999 • Good et al. 1999 • Pazzani 1999 • Billsus & Pazzani 2000 • Tran & Cohen 2000 • Melville et al. 2002	Combining content-based and collaborative components by; • Incorporating one component as a part of the model for the other • Building one unifying model Representative research examples: • Basu et al. 1998 • Condliff et al. 1999 • Soboroff & Nicholas 1999 • Ansari et al. 2000 • Popescul et al. 2001 • Schein et al. 2002

In Table 7, past applications of recommendations systems are briefly proposed (Adomavicius & Tuzhilin, 2005). Moreover, new approaches such as Feature weighting system (Debnath et al., 2008), Luo *et al.* (2008), EMDP (Ma et al., 2007) are presented in detail. Then, *REMOVENDER* (Özbal et al., 2011) is discussed and with the help of these researches our approach offers different point of views that are introduced in the following chapter.

CHAPTER 4

THEORETICAL BACKGROUND

In general, MoresysGOAL is a movie recommendation system using with goal programming language OPL uses the content boosted collaborative filtering approach and with the help of Feature Weighting Content-Based (Debnath et al., 2008), Effective Missing Data Prediction (Ma et al., 2007), Local and Global User Similarity (Luo et al., 2008) studies, considers the data sparsity problem.

MoresysGoal has two parts; these are the loading content information and calculations of the content weights consists of a mathematical model, and the second one is the effective missing data prediction process that aims to provide better movie recommendations to the end users. Content weights, i.e, feature weights are the properties of the movies that have different meanings for the users. For instance, Meg Ryan may be one of the most successful artist for a specific user, or Francis Ford Cappola may be popular director for this user. Thus, director and cast properties of a movie may have different weights for the users. In the following section, computation of feature weights are discussed in detail.

4.1. Calculation of Content Weights

Weigths calculated in the proposed study (Debnath et al., 2008) algorithm are not used in this thesis. This thesis offers a different way for calculation of the feature weights. Instead, goal programming is used for computing content weights. For this purpose, (IBM's OPL 6.3, 2012) mathematical model software which includes CPLEX 12.1 solver is used for neccessary calculations. OPL stands for Optimization Programming Language and is used for finding optimal solutions for proposed linear and integer programming model.

4.2. Mathematical Modeling and Optimization

A mathematical model is a representation of a challenge of interest and is a relevant component to the process of solving that challenge in a optimal way (Sarker & Newton, 2008). Since real problems are modeled by mathematical representations difficultly, numerous assumptions and approximations may be provided by the researchers. Models are the tools used for solving design, managerial, and planning challenges in which decision maker should allocate limited resources among several activities for making a measurable goal optimization (Winston & Goldberg, 2003). In the following section, the different components of a mathematical model are introduced in detail.

4.3. Components of a Model

Mathematical model has three main parts: decision variables, objective function and constraints. In brief, these components are discussed in the following sections.

4.3.1. The Decision Variables

The decision variables are under control of the decision makers, and they influence the whole system's performance. They are used for deciding maximizing or minimizing the values of the decision variables for an objective function (Sarker & Newton, 2008).

4.3.2. The Objective Function

The objective function indicates the goal of the problem by means of using decision variables. Information such as profit or cost per each item are variables required in interaction by decision variables forming the objective function and these variables are also known as objective function's coefficients (Sarker & Newton, 2008). In addition, more than one objective function may be neccessary for an organization in

many conditions. For instance, the Monroe County School Board in Bloomington, Indiana anounced that the students's assignment considers the objectives as below (Winston & Goldberg, 2003).

- *The number of students at the two high schools are equal to each other*
- *The average travelling distance of students minimizes*
- *Student body at the two high schools has diversified*

Furthermore, there are also problems that have only one objective function as seen in this study.

4.3.3. Constraints

The constraints are the limitations of the values of decision variables. A constraint has two parts, a constant and a function that are rely on either an equality or inequality sign (Sarker & Newton, 2008). The function represents the total resource needed by means of decision variables and the constant means the avaliability of the total resource in case of a resource constraint.

4.4. Goal Programming

Goal programming aims to find a convenient solution based on the relative importance of each objective. For instance, politicians promise to electors that the debt will decrease in the country. At the same time, they might claim that they will offer income tax relief. In these kinds of conditions, it is difficult to find a single solution for optimizing these two multiple objects (Taha, 2007).

In the following sections, two different methods are presented for solving goal programming. For a single objective function, the multiple goals are provided in both methods.

4.4.1. The Preemptive Method

The decision maker should rank the problem's goals in importance order in the preemptive technique. The objectives of the problem are stated as below for a n goal condition (Taha, 2007)

$$Minimize\ G_1 = p_1\ (Highest\ priority) \tag{4.1}$$

$$Minimize\ G_n = p_n\ (Lowest\ priority) \tag{4.2}$$

where p_1 is the part of the deviational parameters, s_1^- or s_1^+ , that indicates first goal. The operation of the solution concerns in one goal at a time, G_1 means the highest priority in the beginning and G_n is the lowest priority at last. This thesis is related to the regression method that does not need to the priorities. Thus, *The Weights Method* (Taha, 2007) is reasonable way in this thesis, is discussed in detail later.

4.4.2. The Weights Method

The Weights Method uses a single objective function obtained as the weighted aggreagation of the functions that indicates problem goals (Taha, 2007). It can be assumed that *ith* goal of *n* goals in the goal programming model is stated as

$$Minimize\ G_i, i = 1, 2, 3, \dots, n \tag{4.3}$$

Then, in the *Weights Method* combined objective function is introduced as;

$$Minimize\ z = w_1 G_1 + w_2 G_2 + \dots + w_n G_n \tag{4.4}$$

where variables $w_i, i = 1,2,3, \dots, n$ are positive weights that effects the preferences of the decision maker dealing with the relative importance of each goal. In the light of the above explanations, variables of our mathematical model are shown as follows

Set & indices

m,n : movies, m,n $\in$ *Movies =* $\{ m_1, m_2, ..., |Movies| \}$

f,k : features f,k $\in$ *Features =* $\{f_1, f_2, ..., |Features|\}$

Parameters

$a_{f,m,n}$ *: value of attribute f between movies m and n*

$s_{m,n}$ *: similarity between movies m and n*

Decision variables

w_f : weight of feature f, $w_f \geq 0$, $w_f \in (-\infty, \infty)$

d^+_{mn}: positive deviation for interaction in movies m and n

d^-_{mn}: negative deviation for interaction in movies m and n

Mathematical Model **(1)**

$$Minimize \sum_{m=1}^{|Movies|} \sum_{n=1}^{|Movies|} (d^+_{mn} + d^-_{mn}) \tag{4.5}$$

subject to

$$s_{m,n} + d^+_{mn} - d^-_{mn} = \sum_{f=1}^{|Features|} w_f \, a_{f,m,n} \tag{4.6}$$

where $\forall$ m $\in$ Movies, $\forall$ n $\in$Movies, m < n

$$w_f \text{ , } unrestricted \ \forall f \in features \tag{4.7}$$

$$d^+_{mn}, d^-_{mn} \geq 0, \forall m \in Movies, \forall n \in Movies \tag{4.8}$$

First, mathematical model considers unrestricted feature weights. In other words, positive of negative values are convenient for linear model. With the take account of constraint 5, the mathematical model only considers the positive weight values.

$$w_f \geq 0 \tag{4.9}$$

At last, constraint 6 that means aggregation of the weights of the features is equal to 1, is inserted to the mathematical model.

$$\sum_{f=1}^{|Features|} w_f = 1 \tag{4.10}$$

Primary purpose is to make curve fitting, which aims to optimize errors both negative and positive perspective and then tries to fit the curve. First of all, constraints are identified such as defining first movie to the last movie these values are written but if there is a *1->2, 2->1* is ignored.

Table 8. Identified Constraints (First Movie 1 to The Second Movie 2)

1->2	2->1	3->1	4->1	…		
1->2	2->2	3->2	4->2	…		
1->3	2->3	3->3	4->3	...	...	
...	...	…	4->4	…		
1->1681	2->1681	3->1681	…	…	1681>1681	1682>1681
1->1682	2 ->1682	3->1682	4->1682	...	1681> 1682	1682>1682

As a result of values seen above Table 8, there remains only one record for *1682->* because all the others are identified past. Therefore, the result of *1+2+3+...+1682* summation is the number of constraints. Based on gauss sum formula, then, *(1682*1683) /2 = 1.415.403* is the obtained constraints within each different pairs. When OPL is run for this huge size constraints, there exists OutOfMemory error. Neither OPL binary nor servers of laboratory can calculate this model. Although these constraints are sufficient to calculate weights, due to obtaining more reliable and effective weights, movies in which users gave more than one hundred ratings are selected in relevant database table in MSSQL. As a result of this, number of constraints reduce to *338 * 339 / 2 = 57291*.

$$\text{Similarity}[i,j] = w_1 * F_1[i,j] + w_2 * F_2[i,j] + \ldots + w_7 * F_7[i,j] \tag{4.11}$$

In this formula, between F_1 and F_7**,** which are all the feature vectors**,** are taken from feature content table. Lastly, the remaining part of the formula is similarity matrix can ben calculated by item based pearson correlation in Mahout.

$$ActItemSimilarity[m,j] = \frac{\sum_{u=1}^{U}(R_{u,m} - \overline{R_m})(R_{u,j} - \overline{R_j})}{\sqrt{\sum_{u=1}^{U}(R_{u,m} - \overline{R_m})^2 + \sum_{u=1}^{U}(R_{u,j} - \overline{R_j})^2}} \quad (4.12)$$

Actual Item Similarity matrix is calculated from first movie to the 338. (last) movie in Mahout Eclipse application. Pseudo code of this calculation is stated below;

For(Movie i in the 338 movie list)
For(Movie j in the 338 movie list)
ItemSimilarity = Item Based pearson in Mahout (Formula 10)
Calculate ItemSimilarity(i , j)
Each calculated value is written in text file

At last stage, actual similarity matrix is obtained in a single text file. Writing to the text files is easy and fast way and conversion of this type text files to database and excel format is also easy. Because datasets studied in OPL must be one of these kinds of file formats. Actual similarity matrix has 338 rows and 338 columns, square matrix and then other feature matrixes have to be square matrix same size as actual similarity matrix.

By implication, feature vectors are converted to 338x338 matrixes. As a result, there are eight 338x338 square matrixes that are retrieved by Excel worksheets and then calculation of seven weights are started in OPL Studio 6.3.

In the first step,based on the assumption that there is no more constraints and the weights are calculated as below;

w = -0.0044519, 0.035238, 0.069244, 0.078955, 0.059107, 0.13703, 0.46915

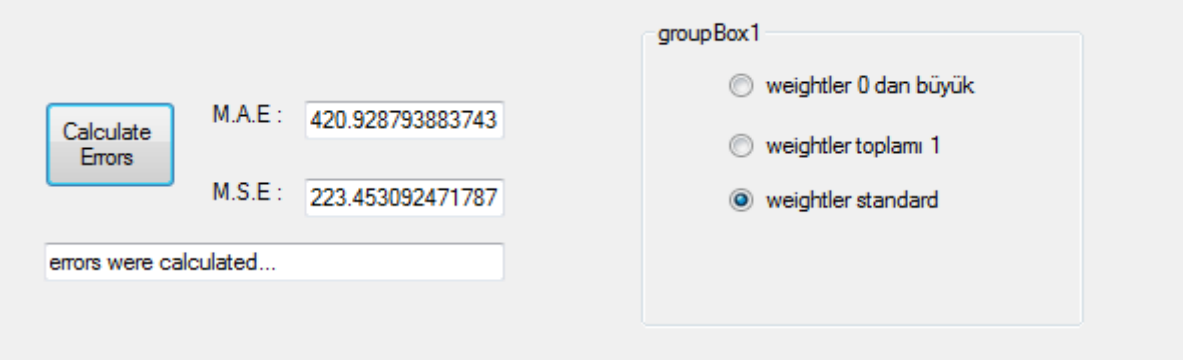

Figure 2. The Standard Calculation of the Weights

These weights show that there is a negative effect of the first year feature's weight and country's weight has the biggest effect to item similarity calculation. Then, a small *c#* desktop application is written to calculate Mean Absolute and Mean Square errors as seen in Figure 2 and 3. These are not relevant to the evaluation phase these calculations only give a comparison of three different conditions of weights. Next two situations are experimented for the proposed study (Debnath et al., 2008). Because in this paper, weights are positive values and their summation is equal to 1. Next step is only considered the summation of weights which is equal to 1 condition, and then obtained weights are;

$$w = -0.0045327, 0.035771, 0.068752, 0.079115, 0.058761, 0.14572, 0.61642$$

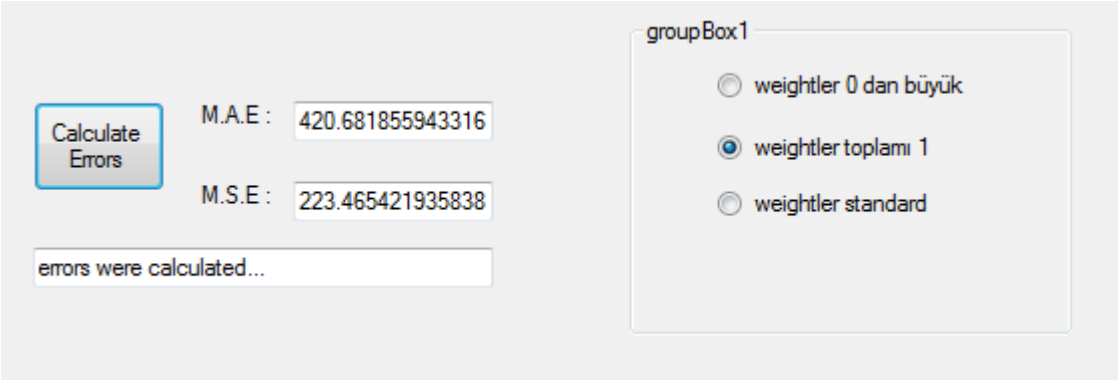

Figure 3. The Summation of the Weights is Equal to 1

In Figure 4, one more constraint that the last situation gives is that weights are positive numbers. Thus, calculated weights are;

$$w = 0, 0.03568, 0.068269, 0.075411, 0.058099, 0.14564, 0.6169$$

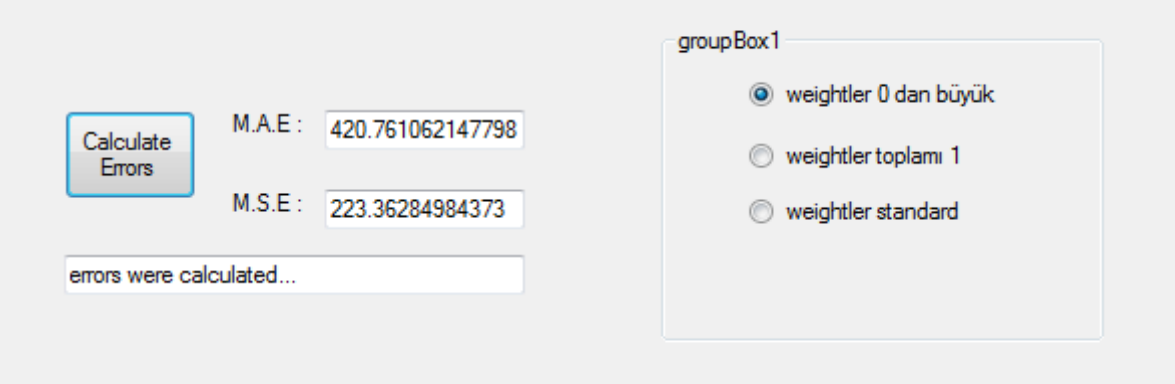

Figure 4. The Weights are greater than 0

In the first step, there are 57291 constraints. Moreover, two new constraints mentioned above are inserted to the list of existing constraints. In the new situation, there are 57293 constraints. It should not be affected by the overall score because there exists huge number of constraints at first. MAE and MSE errors prove this assumption because there are only few differences in decimal portion of the values.

Lastly, all values are known of the right part in Equation 11. All feature vector matrixes are calculated before starting weight measurement in OPL from first feature vector $F_1[1682,1682]$ to the last feature vector $F_7[1682,1682]$. Then, these values are multiplied by the first obtained weights and then similarity matrix was predicted;

$$\begin{aligned} \text{PredSim}[i,j] = & -0.0044519 * F_1[i,j] + 0.035238 * F_2[i,j] \\ & + 0.069244 * F_3[i,j] + 0.078955 * F_4[i,j] \\ & + 0.059107 * F_5[i,j] + 0.13703 * F_6[i,j] \\ & + 0.46915 * F_7[i,j] \end{aligned} \tag{4.13}$$

In the small desktop application which is written in C#, predicted similarity matrix is also converted in a text file for the purpose of easy and fast calculation process. Because this predicted similarity matrix is used for missing data prediction. In the following section missing data prediction approach will be explained.

4.5. Missing Data Prediction

User movie matrix has a very sparsity dataset. Each user could watch each movie, there should be *943 users * 1682 movies = 1.586.126* ratings in this dataset. But user movie rating matrix consists only *100.000* of them. This small calculation expose the huge sparsity problem. At the beginning, Table 9 gives an idea of this sparsity problem

Table 9. User Movie Matrix (UMR)

Movies / Users	1	2	3	4	5
1	$R_{1,1}$	$R_{1,2}$	0	0	0
2	$R_{2,1}$	0	0	$R_{2,4}$	0
3	0	0	$R_{3,3}$	0	0
4	0	0	0	0	$R_{4,5}$
5	0	$R_{5,2}$	0	0	0

In addition, missing data formula is handled by using Equation 12 in the prediction similarity matrix. First of all, for each movie the nearest two neighbors are found out in prediction similarities. Then, item based rating prediction is done

$$ib_{r_{m,2}} = \frac{\sum_{i=0}^{2}(predictedSim(i,j) * actualRating(m,j))}{\sum_{i=0}^{2} predictedSim(i)} \quad (4.14)$$

After missing data prediction process is done, new updated user movie rating matrix (UMR) has appeared as seen in Table 10 below

Table 10. Updated User Movie Matrix (UMR)

Movies / Users	1	2	3	4	5
1	$R_{1,1}$	$R_{1,2}$	0	$\widehat{R_{1,4}}$	$\widehat{R_{1,5}}$
2	$R_{2,1}$	$\widehat{R_{2,2}}$	0	$R_{2,4}$	0
3	$\widehat{R_{3,1}}$	0	$R_{3,3}$	0	$\widehat{R_{3,5}}$
4	$\widehat{R_{4,1}}$	$\widehat{R_{4,2}}$	0	$\widehat{R_{4,4}}$	$R_{4,5}$
5	0	$R_{5,2}$	$\widehat{R_{5,3}}$	$\widehat{R_{5,4}}$	0

In this sample layout, whereas first density of the user movie rating matrix is *7 / 25 = 28%*, at last density will increase to *17 / 25 = 68%*. Density could not be *100%* because this algorithm does not try to predict all missing data. It has just considered to predict rating for an actual movie with item similarity values of the nearest two neighbors. Furthermore, predicted similarity gives the two closest resemblance for the actual movie. From now on, user based pearson correlation that is the last point of the whole system, is discussed next section.

4.6. User-Based and Item-Based Collaborative Filtering Approach

After missing data prediction process, updated user matrix is the input for user based pearson correlation. Previously, item based pearson correlation is used to

calculate actual item similarities. At this moment, User-based Pearson Correlation concerns the similarities showing previous movie rating values belonging to the users, and is used from Equation 13 in the proposed study (Owen et al., 2011). $P_{a,u}$ user similarity of a and u users then is calculated by;

$$P_{a,u} = \frac{\sum_{m=1}^{M}(R_{a,m} - \overline{R_a})(R_{u,j} - \overline{R_u})}{\sqrt{\sum_{m=1}^{M}(R_{a,m} - \overline{R_a})^2 + \sum_{m=1}^{M}(R_{u,m} - \overline{R_a})^2}} \quad (4.15)$$

Where $R_{a,m}$ means given rating for movie m by active user a, $\overline{R_a}$ active user's average rating value for common movies with the compared user u. M is the number of total movies. $R_{u,m}$ is the rating of the movie m given by user compared user u.

Therefore, last stage of the recommender system is completed. Recommended movie lists with a rating prediction is provided for the users. For already-existing users and movies all the prediction processes are made in Mahout and then are transferred to the MSSQL database. In the following section will be related to the impact of the parameters to the system.

4.7. Impact of Thresholds

Initially, this section gives an explanation about parameters that are used in other studies in the literature mentioned above. Then, how parameters influenced MoresysGOAL is also explained.

4.7.1. Impact of Threshold α

The first step is to check parameter α mostly issued in which study, and then it is observed that it is mostly related to the proposed article (Luo et al., 2008). It is an important mechanism for rating prediction phase in this approach. If it is set to 0, rating prediction is only related to local user similarity that provides more distinctive information about a movie which has less common ratings. In other case, when it is set to 1, rating prediction is completely concerned in global user similarity.

The parameter α aims to give comprehensive improvement for the user-based similarity and in their approach, it is equal to 0.5. Due to the consideration in the study (Luo et al., 2008), α is ignored because there is sufficient amount of ratings for actual users in MoresysGOAL during successful content-based rating prediction process and it also wants to indicate power of content based similarity prediction mechanism.

4.7.2. Impact of Threshold β

In the proposed study (Özbal et al., 2011), β parameter is the *importance weighting* of content similarity over prediction process. In addition, minimum value of it is 0 and maximum value is 1. If it is equal to 0, predicted item similarity is only related to Mahout item-based similarity. Whether β is set to 1, on the other hand, item similarity completely relies on content similarity. As a result of providing under similar circumstances in the article (Özbal et al., 2011), MoresysGOAL takes *0.5* value for β.

The parameter β has a critical mission for showing the power of the calculation of content similarity weights. When Ozbal *et al.* (2011) take content weights from the literature study (Debnath et al., 2008), MoresysGOAL obtains these content weights from curve fitting process by creating linear model in OPL.

4.7.3. Impact of Thresholds γ and δ

In the three studies (Özbal et al., 2011), (Ma et al., 2007) and (Luo et al., 2008), γ equals to 30 and it is used for calculating user-based similarity values to determine minimum commonly rated items by two users. This parameter is essential to protect these users from similarity devaluation. These users may not have common tastes, but there may be a similarity overestimation if they have given ratings to few movies.

MoresysGOAL uses some kind of different version of significance weighting for both users and items. In Mahout collaborative filtering using Pearson collaboration correlation, a term is employed for weighting. It provides a quick response to the correlation to push towards 1.0. If there are fewer correlations of movies, otherwise, it is pushed towards to -1.0 where more movies are under circumstances in which users have satisfactory amount of rated movies than γ and δ values are automatically changed in Mahout. Therefore, workload of calculation of each pairs in each step can be decreased

and MoresysGOAL can give quick response by the help of significance weighting implementation served in Mahout.

4.7.4. Impact of Thresholds η and θ

Initially, η and θ set to 0.5 in the article (Ma et al., 2007); they have both set for collaborative filtering approach. While η is threshold value of user based neighbor selection, θ is the parameter of selecting item-based neighbors. If each parameter equals to 0, this approach could perform all missing data prediction. Otherwise, if they are all set to 1, their application cannot make missing data prediction as seen in Table 11. Therefore, the density of user movie rating matrix could be increased.

Table 11. The Relationship of Parameters
(Source: Ma et al., 2007)

Related CF Approach	***β***	***η***	***θ***	***λ***
User-based CF without missing data prediction	1	1	1	1
Content and item-based CF without missing data prediction	0	1	1	0
User-based CF with all the missing data prediction	1	0	0	1
Content and Item-based CF with all the missing data prediction	0	0	0	0

In other words, the main problem in this situation is that, density which is increased during the missing data prediction is how to reflect the overall accuracy. Both of these variables are equal to each other at the beginning of the evaluation process, because it is easier to control their significance.

4.7.5. Impact of Threshold λ

In similarity fusion approach (Wang et al., 2006), λ parameter determines how the correlation is selected from either users or items. If λ is equal to 0, their approach is exactly an item-based approach and whether it is set to 1, their study is only related to the user-based Pearson correlation.

Table 12. The Relationship of Parameters in the Last Situation

Related CF Approach	***α***	***β***	***δ***	***γ***	***η***	***θ***	***λ***	***k***
EMDP			25	30	0.5	0.5	0.7	
SF			0.7				0.7	35
LU&GU	0.5			30				35
SCBPCC							0.35	20
GO_CBCF	0.5	0.5	25	30	0.6	0.6	0.6	
MoresysGOAL		**0.5**	**25**	**30**	**0.5**	**0.5**	**0.5**	

Table 12 gives a summary for all parameters used in the studies. There is a comparison between MoresysGOAL and other studies proposed in Table 12. In the proposed research (Ma et al., 2007), λ is also used for different configuration that considers selection of either collaborative filtering or cluster-based smoothing. As a result, MoresysGOAL uses λ threshold as experimented in the former SF approach, because system design of MoresysGOAL does not include cluster-based smoothing. Thus, it predicts the ratings within consideration of collaborative filtering either user-based or item-based. The parameter λ is initially set to 0.6 in the studies (Özbal et al., 2011; Ma et al., 2007) and then it is also equal to 0.6 in MoresysGOAL. This value mentions that predicted ratings are approximately average of user-based and item-based collaborative filtering but it is a little close to the former.

CHAPTER 5

SOFTWARE ARCHITECTURE

The aim of this chapter is to provide recommended lists to the users in graphical user interface. First step is the calculation of weights, and then with the help of these weights, the predicted item similarities are calculated. Later, these predicted item similarities are used to predict missing data process, and finally collaborative filtering approach uses this updated dataset to give user recommended lists.

As shown in the system architecture the main aim of the system is to give predicted rating list to the online users in descending order. This chapter concerns in the software design and information extraction processes. There are three main applications in the processes of information extraction that are introduced in the following sections.

5.1. System Architecture

System architecture given Figure 5 summarizes the whole picture of MoresysGOAL design. Preparation of the inputs, obtaining the recommended list and graphical user interfaces are the three main parts of the system. In terms of preparation these are;

1. **Content Information Extraction:** Contents are loaded from IMDb to local MySQL database. Also, with the help of the ASP.NET web application movie contents are stored into the MSSQL database. Both of the databases are constructed and then they are ready to be inputs of the Recommendation Engine.
2. **Recommedation Engine:** This part is the core component of the system. It consists of the calculation of content weights. Therefore, several prediction processes are started to be given a recommended list for users.
3. **Graphical User Interface:** Users can see their own recommended list in these interfaces. Also, they can search a specific movie, give rating to this

specific movie, etc. Moreover, administrator can make some operations such as updating both databases and recommendation processes.

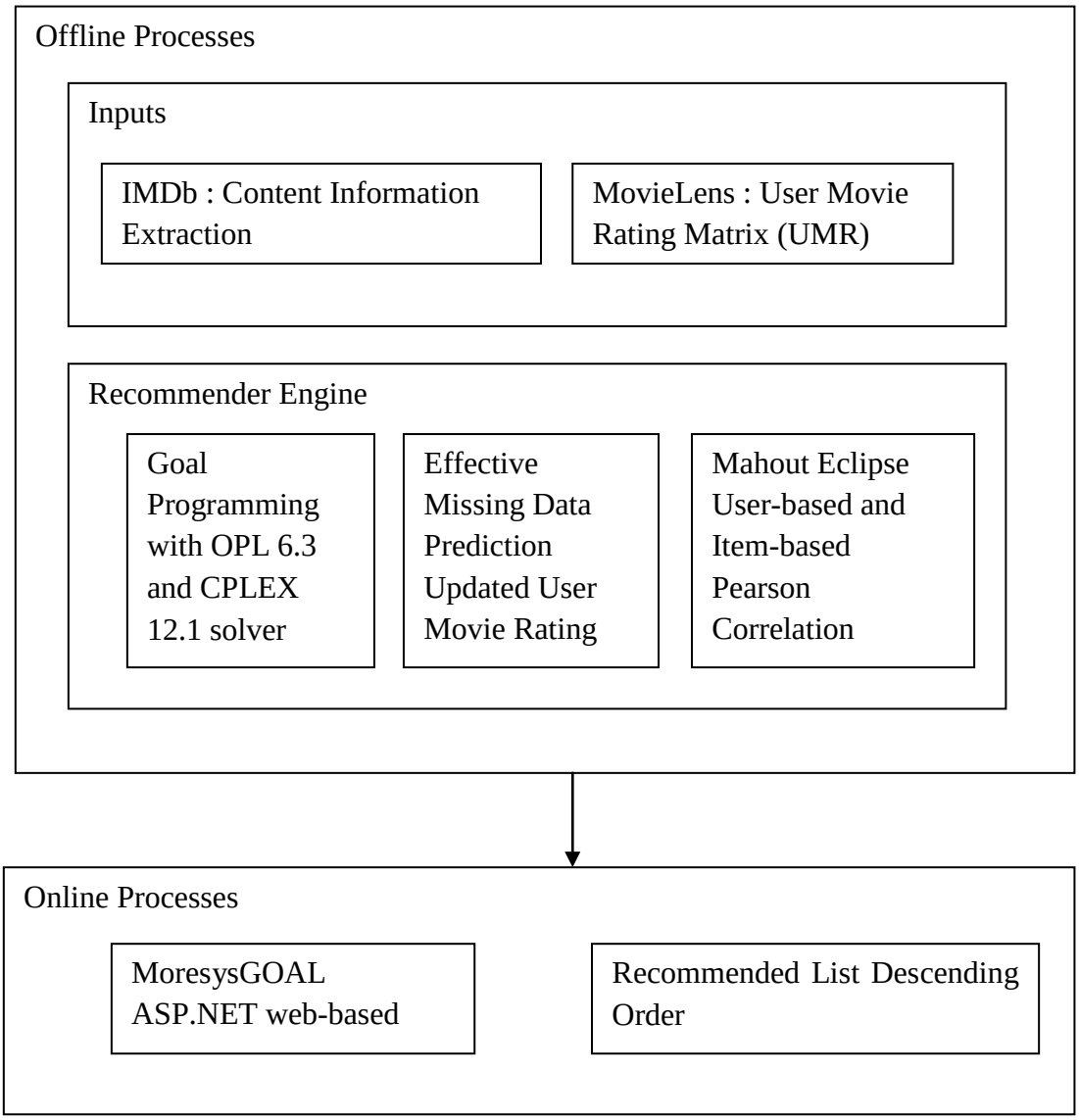

Figure 5. Software Architecture of MoresysGOAL

5.2. Content Information Extraction

As seen in Figure 5, content information is extracted IMDb movie database by using both MySQL 5.2 CE and MSSQL 2008 R2 in two ways. The first way is to use (IMDbPY, 2012) which can retrieve IMDb dataset from the plain text files and can be stored in MySQL. IMDbPY is written in python programming language and is open source package. For this reason, plain texts of IMDb are downloaded in Ubuntu 10.04

operating system. After this, IMDbPY package is run to retrieve dataset to MySQL database.

In the second way, an ASP.NET web application is written in C# language for retrieving IMDb dataset to MSSQL database. Therefore, a cross check can be made for both datasets and it can give more clear and reliable input data to the prediction phase.

5.2.1. MySQL Database Structure

MySQL database is obtained by the information extraction techniques. MovieLens dataset has a unique attribute that is a combination of movie title and year properties, whereas IMDb local database has both title and year attributes. MovieLens dataset is separated into two attributes for the purpose of the joining MoviLens and IMDb local databases. Because they have only these common two attributes.

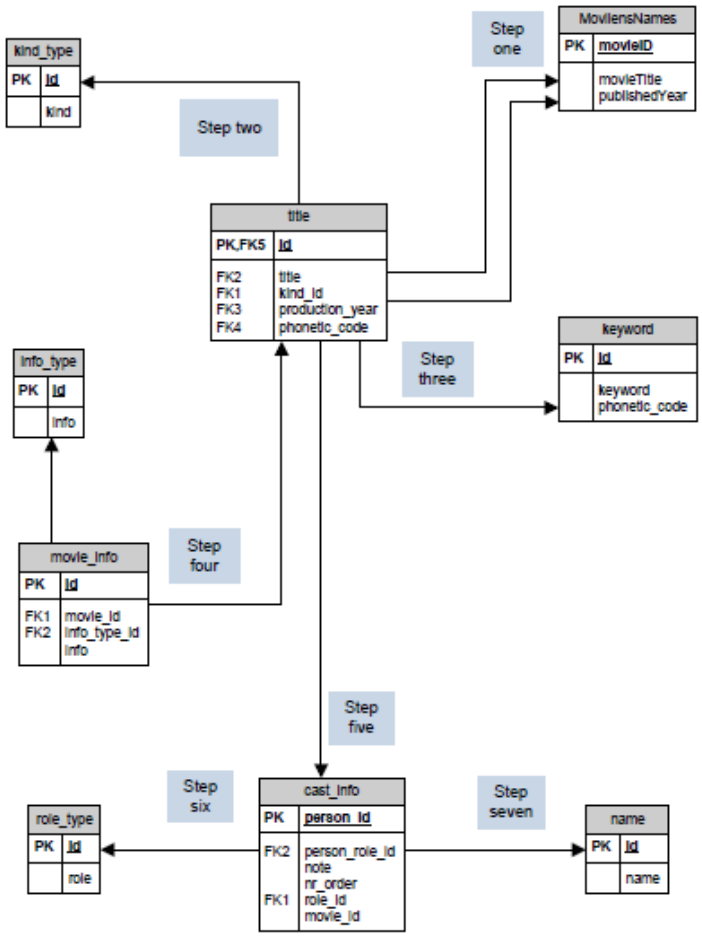

Figure 6. Database Schema of IMDb Database

IMDb local database has huge size tables. As a result of this, join queries are written in mind so as to reduce the size of joining tables. All seven steps shown in Figure 6 aim to reduce size of the last *select* query. In the first step, 1682 movies from the MoviLens table join to the *title* table of IMDb local database. *Title* table has more than two million records. Hence, only 1682 movies are retrieved *title* table. Thus, kind attribute of the *kind_type* table and keyword property of *keyword* table produce data of 1682 movies are selected in a very short time.

Step four has several content information such as genre, country, language, etc. and this step took relatively longer than time than first three steps as seen in figure 2. However, the first join step also provided minimization for the process time of step four.

Last three join steps need to retrieve actors and actress list of the movies. Furthermore, name table of the IMDb local database consists of writers' names. Nevertheless, these processes have the longest time period, because of the huge sizes of both *cast_info* and *name* tables.

Extraction information from the tables is not the first problem. There is also another one that occurs in the source engine types of MySQL. Firstly, InnoDB source motor is attempted but it took almost 20 gigabyte size in the hard disk. Hence, there exists a gigantic size file which is ibdata1 in the *ProgramData* directory in Mysql main directory.

Therefore, MySQL source engine is converted to MyISAM. After this conversion, attributes which are essential for the joining tables are indexed. It is really effective and fast in both space and time constraints.

In the step one, out of 1682, 500 movies are left which are not in the IMDb local database. It is so hard to search each movie and then retrieving their contents. As a result of this, a web based ASP.NET application is written to overcome these difficulties. The following section gives detailed information about ASP.NET web application.

5.2.2. ASP.NET Screen Scrapper Web Application

Web application is written in C# programming language in Visual Studio 2010. When title and year attributes joining of Movielens and IMDb tables are done, there are

500 movies that are not in the IMDb local database. ASP.NET web application is developed to find missing movies, then is used to give detailed content information to the users in graphical user interface.

The interface of web application has a simple table, because results must be shown fast while making screen scrapping. Screen scrapping is a method supposing that a user searches the movie in Google search, than clicks the link IMDb internet site is opened, then clicks the right mouse button, presses view page source and takes the html script text, finally picks the relevant contents for each movie.

Actually, it will be an illegal way of retrieving data, if it is a commercial manner. However, it is used for only academic purposes. Because of this, overall system will not be an online application. Meanwhile, it may not be effective way of collecting the whole movie information with this method. Some errors such as internet connection timeout and null reference pointer may occur. Five hundred movies can be retrieved almost 2 hours and it took 2 days for all 1682 movies.

First of all, a copy of MovieLens database table is loaded to MSSQL Server 2008 R2. As a result, it is easy to connect Visual Studio 2010 to MSSQL. For this purpose, *ml_movies* table is created in MSSQL. Actually, this created table is the same as the item dataset in the 100K zip file of Grouplens.

In the MSSQL movie titles retrieved from *ml_movies* are used in ASP.NET web application to get movie contents by IMDb internet site. When each movie content data is taken, feature contents which are nine attributes with seven features, *imdbid* and *movieid* attributes of each movie are stored in *tblimdb* table in MSSQL. Therefore, 1682 movie content information are copied and these are ready to start content boosted process.

Moreover, the difficulties when the content databases are created, there occurs character encoding problems. Movies in the MovieLens dataset are all around the world from China to USA. Cultural differences may cause some problems, when writing names of the writers or actors. This type of character encoding problems are handled with the help of server html encode and decode functions that are provided in ASP.NET. UTF8 encoding is used for common language character encoding.

In addition, another difficulty is working with server of IMDb site. Because it has a short timeout limit, it may be made to protect the site by illegal data retrieving. Timer is added to start the application automatically whether some of the features have

null values. The input dataset is obtained to use content-boosted algorithm and, is discussed in detail in the following section.

5.3. Database Structure

MovieLens dataset belonging to GroupLens research group is examined that there are many practices with this dataset. User movie rating matrix that is obtained by MovieLens is used for collaborative filtering. Also, content information about these movies collected from IMDb is used for content boosted technique.

5.3.1. Content Features

While obtaining content of the movies, feature vectors are necessary for each movie. Seven features are selected from IMDb local database; year, rating, genre, language, country, writer and cast. Furthermore, these features are used to show the content of a specific movie in graphical user interface. Year and rating features have discrete values, at the same time; other attributes have many values as seen in table 1.

Prediction techniques are normalized to 0, 1 range. For this purpose, similar distance measures are used that are passed of FWCB (Debnath et al., 2008) study. Year, rating features are first subtracted and then divided by an unreachable number. Other features have string values more than one, intersection of these features divided to the feature that has minimum string list length in this intersection.

Debnath *et al.* (2008) offers such an operation is done by dividing by the feature that has the maximum string list length. For instance, genres of *A* movie are drama, romance, action; genres of *B* movie are Action, Comedy, Sci-Fi; and finally genres of *C* movie are action and horror.

While distance measure of *A* and *B* movies is calculated, common genre is action in both movies divided by string lists of *A* or *B* because both have 3 string list length. As a result, distance measure of these movies is equal to 0.33. Likewise, *A* and *C* movies have one common genre *Action* is then divided by 2 that is minimum size genre list of *C* movie. If this assumption is used, the result is equal to 0.5. But if FWCB paper is used to calculate distance measures, this time the result will be 0.33 and *B*, *C* movies

have same similarities to a movie. With this new distance measure assumption, similarity values are improved and they are listed in Table 13.

Table 13. The Distance Measures of Features

NO	Feature	Type	Domain	Distance Measure
1	Year	Small Integer	[1900, 2012]	$\frac{300 - \lvert y_1 - y_2 \rvert}{300}$
2	Rating	Small Integer	[1,5]	$\frac{10 - \lvert r_1 - r_2 \rvert}{10}$
3	Genre	String List	Action, Comedy, etc.	$\frac{g_1 \cap g_2}{g_1}$
4	Language	String List	English, Mandarin, etc.	$\frac{l_1 \cap l_2}{l_1}$
5	Country	String List	USA, UK, etc.	$\frac{co_1 \cap co_2}{co_1}$
6	Writer	String List	Francis Ford Coppola, Oliver Stone, etc.	$\frac{w_1 \cap w_2}{w_1}$
7	Cast	String List	Michelle Pfeiffer, Al Pacino, etc.	$\frac{ca_1 \cap ca_2}{ca_1}$

5.3.2. User Movie Rating (UMR) Matrix

Prediction technique consists of two parts; content-boosted and collaborative filtering. The former is obtained by the table given above. The latter is downloaded from Grouplens research web site. For a long time GroupLens research group has collected rating datasets. MovieLens 100K dataset that has 100000 ratings by almost 1000 users for 2000 movies is selected for preparing user movie rating matrix. In the future, for improving scalable problems MovieLens 1M and MovieLens 10M will also be examined.

Ratings given each user for each movie are inputs of MovieLens 100K dataset. Rows imply the users and columns are the movies in UMR matrix. There exists *U* users; *M* movies in the *UxM* user movie rating matrix *R*. Ratings have a range between 1 and 5. Maximum value of ratings is 5, minimum value is 1. UMR matrix is converted form of database table and is stored in MSSQL 2008 R2. A copy of content feature tables is also stored in MSSQL. Especially, it makes easy connection process with Visual Studio and MSSQL, because of the preparation of graphical user interfaces in Visual Studio

2010 at the last stage of the system architecture. MovieDB database is created in MSSQL 2008 R2 and following database tables are located in this database

Table 14. The Database Tables in MoresysGOAL

Table	**Attributes**	**Description**
ml_movies	movieID,movieTitle,release_date,imdbURL,Action,Adventure,Animation,Children's,Comedy,Crime,Documentary,Drama,Fantasy,Film-Noir,Horror,Musical,Mystery,Romance,Sci-Fi,Thriller,War,Western	u.item dataset of MovieLens 100K
ml_movieGenre	genreID, genreName	Contains 19 names of genres
ml_userMovieRating	userID,movieID,rating,timestamp	u.data dataset of MovieLens 100K
imdb_movieFeatures	imdbID,movieID,title,year,rating,genre,writer,cast,language,country	Content information of each movie

ml_movies database table has both *movieTitle* and *year* information in a single attribute. This attribute is divided into two parts and then movieTitle and *release_date* attributes are created. Also, genres which are taken from *ml_MovieGenre* database table *genre* are made up of attribute headers of *ml_movies* database table.

ml_userMovieRating table is the main part of the collaborative filtering algorithm. Actually this database table is the main table which the whole system works on. There exists a cross check between *imdb_movieFeatures* table and the query results by joining of MySQL. Thus, all processes about information extraction and databases have been discussed. Next section will focus on recommender components working with these inputs.

5.4. Graphical User Interface

MoresysGOAL, Movie Recommendation System is written in C# programming language in Visual Studio 2010. Also, it is a web based system in ASP.NET Framework

4.0. The main aim of the system is to recommend movies to the users. Moreover, users can give rating to favorite movies and are able to comment on these movies, too. They also have a chance to obtain a general idea of movies also commented by other users. First of all, a login screen appears when you enter the web application.

Figure 7. Login Screen of MoresysGOAL

If the user does not have an account yet, he or she can press to New User button and then write his /her email address and after the password confirmation, they can easily enter the system.

Figure 8. Create New User of MoresysGOAL

After this short login attempt, users can see the main page. In MoresysGOAL users can find the users who have the nearest preferences. Also, they can find the movies which are similar to the previous movies they have seen before. For this purpose, users can search specific movies, too. Then the detailed content such as genre, cast, etc. of the movie can be browsed. Content information consists of *publish year*, *rating, genre, language, country, writers, cast* are the local copies of IMDb database and are related to 1682 movies are taken by Movilens.

Figure 9. Recommended List for a Test User in MoresysGOAL

In Figure 9 above, *firstUser* enters the main page and then can see the most five relevant movies his interests. As soon as users enter the system, they can also see the recommendations which are calculated by prediction approach used in the thesis. This is a top-N style representation which means that movies are shown in descending order. The movie which has the biggest predicted rating is the first member of the list. For instance, *"Batman"* is the most interesting movie for the firstUser according to the users past habits. Further, users can find out the specific movies by clicking the Search link. SearchMovie page then appears in the browser as seen in Figure 10

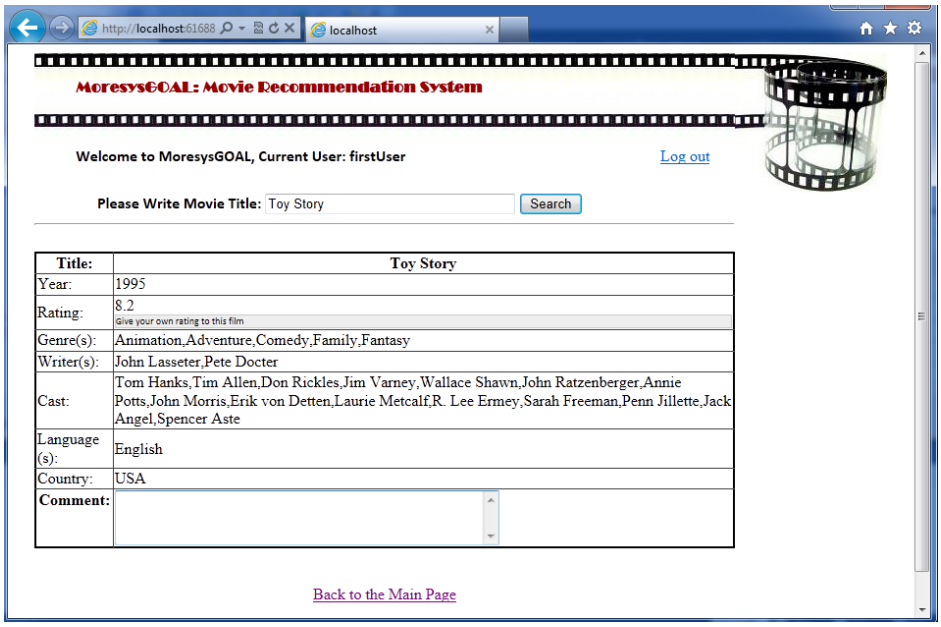

Figure 10. Search a Specific Movie in MoresysGOAL

In Figure 10, first user is interested in Toy Story Film. He can see all seven features of the movie and he can also give his own rating to Toy Story. For this purpose, the first user should press the *"Give your own rating to this film"* button and this popup window is shown below.

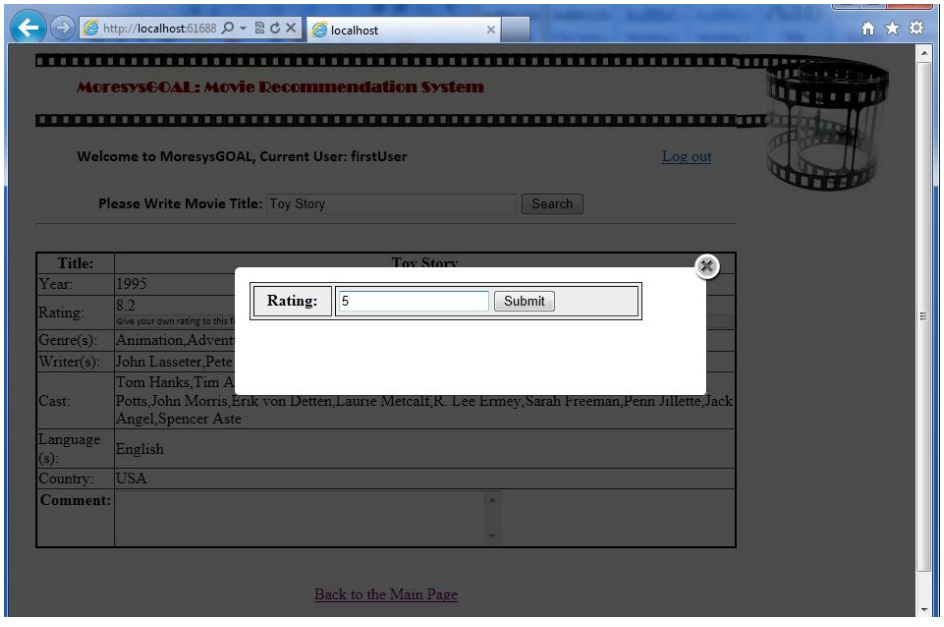

Figure 11. Give an Additional Rating to a Specific Movie

In the above figure, user enters 5 rating to Toy Story movie and click Submit button. These rating values are stored in the database table. After users enter the system,

any day of the week administrator can update to databases taking these new rating values into account.

When administrator logins the system, there are four steps to be followed that are MySQL-IMDb, MSSQL-Movilens, and Content-OPL and Content-UMR links. If there are new movies in the IMDb database, the administrator first clicks the MySQL-IMDb link to up to date IMDb local database.

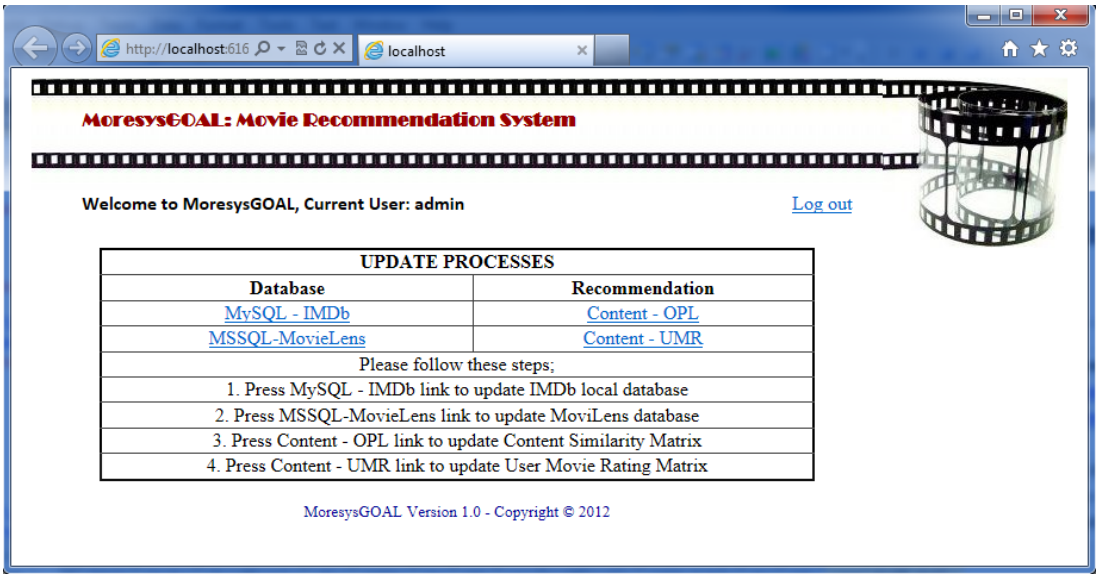

Figure 12. System Administrator Main Page

In Figure 12, if there are new users or movies in MoviLens dataset, the administrator can control this by clicking the MSSQL-MovieLens link and then whether or not all databases in MSSQL is refreshed. If there are significant differences in MySQL content tables, administrator should be attempted to calculate all contents weights again by clicking the Content - OPL link. The last step is the part of the recommendation system is to control updates in the user movie rating matrix. Therefore, all the databases and recommendations of the system can be refreshed and updated by the administrator.

In conclusion, MoresysGOAL is made up of two main parts. The former is the item similarity values that are calculated by content feature weights. These similarities are used for the missing data prediction. Finally, these updated user movie rating matrix is the input of the collaborative filtering approach. The following chapter is all about the evaluation of our approach to the other usages.

CHAPTER 6

EXPERIMENTAL SETUP

Experimental setup process aims to evaluate and test the MoresysGOAL web-based application. The process starts with introducing the dataset and used metrics. Then, there is a comparison between MoresysGOAL and other systems. At last, the prediction performance is discussed by the parameter β determines the extent to which the item similarity relies on content similarity, is explained in the following sections.

6.1. MovieLens Dataset

Experiments are evaluated in MovieLens dataset which is generated by the researchers in GroupLens Research group at University of Minnesota. There are three main datasets in the MovieLens dataset that are hundred kilobytes and one and ten millions datasets. Although one and ten million datasets are able to evaluate scalability, hundred kilobytes dataset is preferred due to encounter problems about memory capacity limitations.

In the hundred kilobytes dataset, it contains 943 users and 1682 movies and 100.000 ratings. Minimum value of rating is 1 and maximum rating value is 5. The proposed studies (Luo et al., 2008; Özbal et al., 2011; Ma et al., 2007) are used based on the assumption that there occurs at least 20 movies rated by each user. Furthermore, in the study (Xue et al., 2005) assumption is that there exists at least 40 movies rated for each user. According to the majority and changing few records, the former assumption is selected then 94968 ratings remained.

Actually, the important point is whether MovieLens dataset will be dense or sparse when the missing data prediction process is handled. Initial sparsity of MovieLens dataset is stated below;

$$Sparsity = 1 - \frac{94.968}{943\ x\ 1682} = 94\ \% \tag{6.1}$$

Equation 1 indicates that sparsity value can be sufficient for the experimentation, because it shows low density as needed. Also, if each user gives a rating for each movie there will be almost more than half a million ratings. It is also mentioned that almost 1.4 millions ratings are neccessary to reach density value to hundred per cent, in other words fully density matrix.

According to the studies which are compared with MoresysGOAL, majority of them prefer the holdout method to evaluate the prediction approaches. In the holdout method, training and testing sets are the two divided parts of the main dataset. After that, majority studies select the 500 random users from the MovieLens dataset. At first, twenty percent, and than fourty percent, lastly sixty percent of this dataset is used for constructing the training sets. Remaining parts of the datasets are also the testing datasets.

In almost half of 1682 movies in the user movie rating matrix, number of rated movies for them are smaller than 20. It remains 939 movies that have been rated more than 20 times. Therefore, the number of user movie rating matrix size is decreased from 100K to 95K. Then 500 movies are randomly selected from this new user movie rating matrix.

In detail, the first 500 users of MovieLens dataset is randomly selected by means of assuming that is at least 20 movies are rated by each user. Second, 20% is used for training and 80% is used for testing (active) users. This dataset is called MovieLens100. Later, MovieLens200 is constructed in the same way, for once 40% is used for training and 60% is used for testing (active) users. The last one is MovieLens300 in which 60% is used for training and rest of them is used for testing users.

Prediction approach uses the training set users to make predictions. Therefore, accuracy of the prediction process is measured by actual users. As a result, sparsity of the actual users varies as if each actual user gives a rating for five movies that will be called Given5. Each test user rated for different ten movies then it is called Given10 and lastly Given20 means there will be twenty movies rated by each actual (test) users.

In brief, total 9 conditions are obtained from the basis of sparsity in both training set and number of movies for each actual user. Due to the fact that, all comparison process is made with other studies (Luo et al., 2008; Özbal et al., 2011; Ma et al., 2007) in similar configurations because they used the same experimental setup. This kind of setup process is preferred because our system should be observed with other literature studies in similar conditions.

6.2. Evaluation Metrics

Mean Absolute Error gives information about average of absolute differences between actual and prediction rating values, and then it converts the negative differences to the positive ones because difference must be higher than zero by dividing the number of the total movies that are voted.

$$MAE = \frac{\sum_{i=1}^{N} |ra_i - rp_i|}{N} \tag{6.2}$$

Where ra_i means the actual rating of the movie *i* and rp_i is the predicted rating value for movie *i*. Then, *N* shows the number of the total test movies. If MAE values are lower, it will show better accuracy and this means predicted ratings are closer to the actual ratings. Instead of taking absolute differences, Mean Square Error takes squares of them whose formula is given below;

$$MSE = \frac{\sum_{i=1}^{N} (ra_i - rp_i)^2}{N} \tag{6.3}$$

Where ra_i is the current rating of the movie i and rp_i is the suggested rating value for movie *i*. The number of the total test movies is *N*. The worst accuracy values demonstrate the highest MSE values.

6.3. Comparison with Literature Studies

General properties of the dataset and the nine conditions of experimental setup are explained comprehensively in the two sections above. Before starting comparison process, validation of each nine configuration is made in a such way that UPCC (User-based Pearson Correlation Coefficient) and MahoutUPCC (Mahout implementation of User-based Pearson Correlation Coefficient) are both traditional algorithms. If mean absolute errors of MahoutUPCC are similar to the errors of UPCC as seen in the evalution setup of the proposed study (Özbal et al., 2011), this dataset will have passed validation control and then it can be ready to be evaluated within MoresysGOAL.

As seen in Table 15, mean absolute errors of MahoutUPCC are close to the values of UPCC in GOCBPCC configuration, so evaluation process is to be done as soon as parameter values are set similar to other literature studies. Initial parameters are selected as $\beta=0.5$, $\lambda = 0.5$, $\eta = \theta = 0.5$ and also number of neighbors is set to 35 due to the fact that these thresholds are similarly set as GOCBCF, EMDP and LU&GU studies. For parameters γ and δ significance weighting is left to consideration of Mahout Weighting. The *weighted* parameter is used in Pearson correlation implementation.

Table 15. MAE Comparisons in MovieLens100 Condition

Number of training users	**MovieLens100**		
Ratings Given for Test Users	**Given5**	**Given10**	**Given20**
MoresysGOAL	0.749	0.764	0.741
REMOVENDER (Özbal et al., 2011)	0.789	0.765	0.756
EMDP (Ma et al., 2007)	0.807	0.769	0.765
SF (Wang et al., 2006)	0.847	0.774	0.792
SCBPCC (Xue et al., 2005)	0.848	0.819	0.789
UPCC	0.874	0.836	0.818
MahoutUPCC	0.847	0.816	0.837

Table 15 shows the mean absolute error values for MovieLens100 condition and for each literature study. According to the order seen in the table, only MoresysGOAL and GOCBCF are specifically related to content information. The rest of all is based on either user-based or item-based collaborative filtering. Also some of them consider the combination of user and item based collaborative filtering. The second one is Ozbal *et al.* content-boosted combination of item and User-based Pearson Correlation Coefficient are called REMOVENDER (Özbal et al., 2011). EMDP stands for Effective Missing Data Prediction for Collaborative Filtering, SF means Unifying User-based and item-based collaborative Filtering Approaches by Similarity Fusion. SCBPCC is Scalable Collaborative Filtering Using Cluster-based Smoothing (Xue et al., 2005). Lastly, MahoutPCC is implemented in Mahout which differs a little from state-of-art user-based pearson algorithm. Moreover, MovieLens200 is the second configuration that contains 200 users for training process and 300 users are dealt with test process stated in Table 16 as follows

Table 16. MAE Comparisons in MovieLens200 Condition

Number of training users	**MovieLens200**		
Ratings Given for Test Users	**Given5**	**Given10**	**Given20**
MoresysGOAL	0.776	0.752	0.743
REMOVENDER (Özbal et al., 2011)	0.782	0.765	0.753
EMDP (Ma et al., 2007)	0.793	0.760	0.751
SF (Wang et al., 2006)	0.827	0.773	0.783
SCBPCC (Xue et al., 2005)	0.831	0.813	0.784
UPCC	0.859	0.829	0.813
MahoutUPCC	0.821	0.809	0.795

In table 16, MovieLens200 configuration indicates that MoresysGOAL gives more accurate results than other literature studies. Furthermore, MovieLens300 situation defines 200 users are test users and 300 users are used for training. Table 17 reveals the MAE scores in MovieLesn300 conditions as seen below

Table 17. MAE Comparisons in MovieLens300 Condition

Number of training users	**MovieLens300**		
Ratings Given for Test Users	**Given5**	**Given10**	**Given20**
MoresysGOAL	0.744	0.739	0.725
REMOVENDER (Özbal et al., 2011)	0.764	0.756	0.738
EMDP (Ma et al., 2007)	0.788	0.754	0.746
SF (Wang et al., 2006)	0.804	0.761	0.769
SCBPCC (Xue et al., 2005)	0.822	0.810	0.778
UPCC	0.849	0.841	0.820
MahoutUPCC	0.854	0.823	0.816

In the tables above, MoresysGOAL is more successful than other studies for each condition due to the improvements on parameters. Instead of using state-of-art user based Pearson correlation coefficient, MoresysGOAL uses effective combination of both user- and item-based approaches. At each nine situations, it outperforms the rest of the studies in the literature seen in the table. It gives smaller error values that mean MoresysGOAL prediction approach increases the accuracy. Especially, when content information is inserted to the calculation of item similarities, it also contributes to the improvement of accuracy. Also β parameter affects the relevance of content similarity as shown figures below

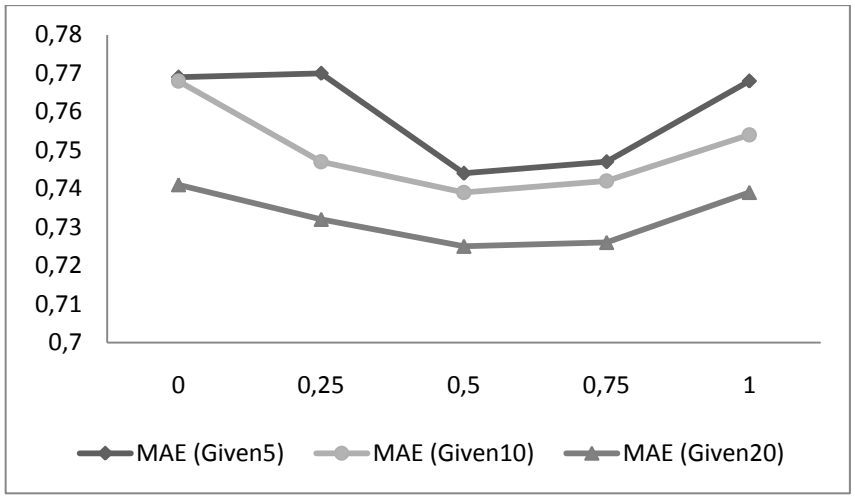

Figure 13. MovieLens300: Impact of β Parameter

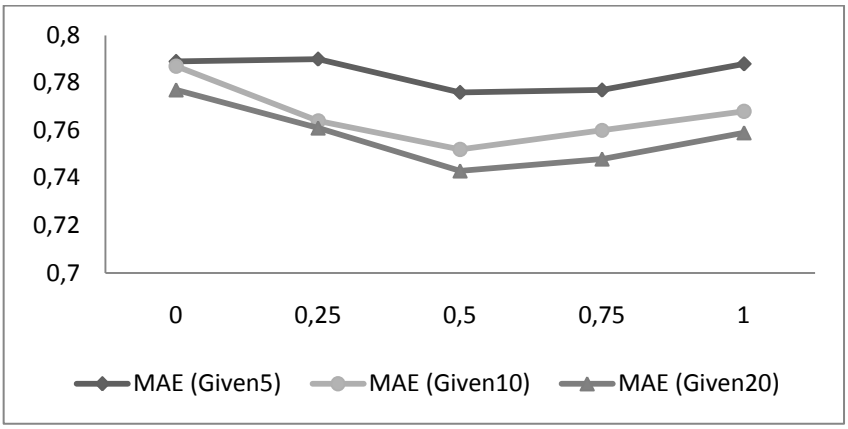

Figure 14. MovieLens200: Impact of β Parameter

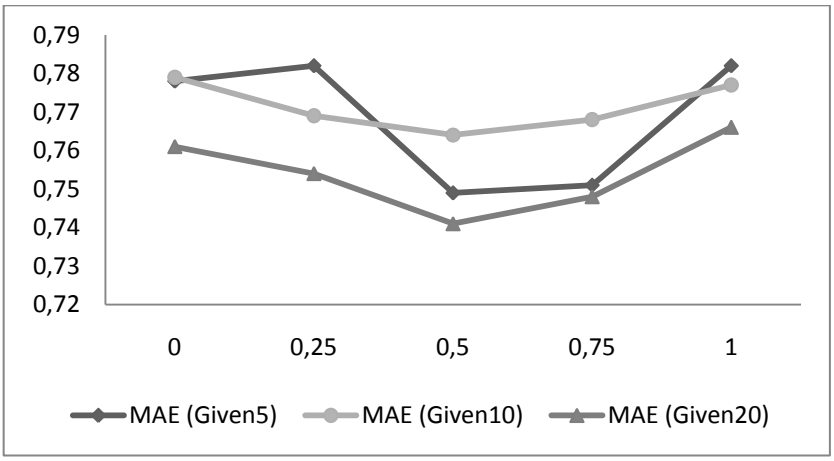

Figure 15. MovieLens100: Impact of β Parameter

In figure 13, it indicates that MovieLens300 condition gives better results than others as given in figures 14 and 15. In the following figures show the relations with the data sparsity and parameter β

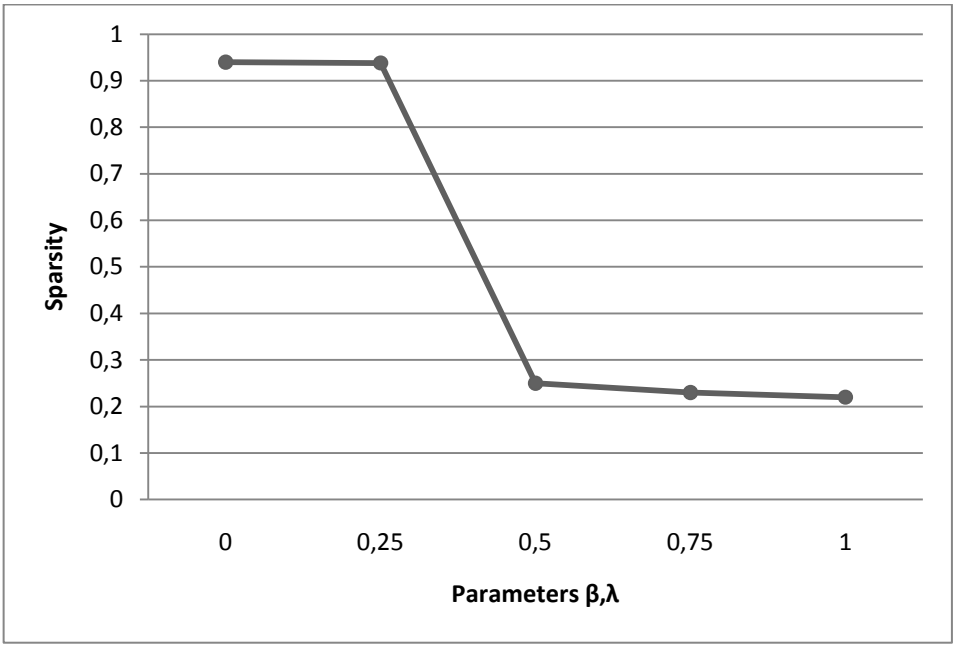

Figure 16. MovieLens300: Impact of β, λ Parameters on Sparsity

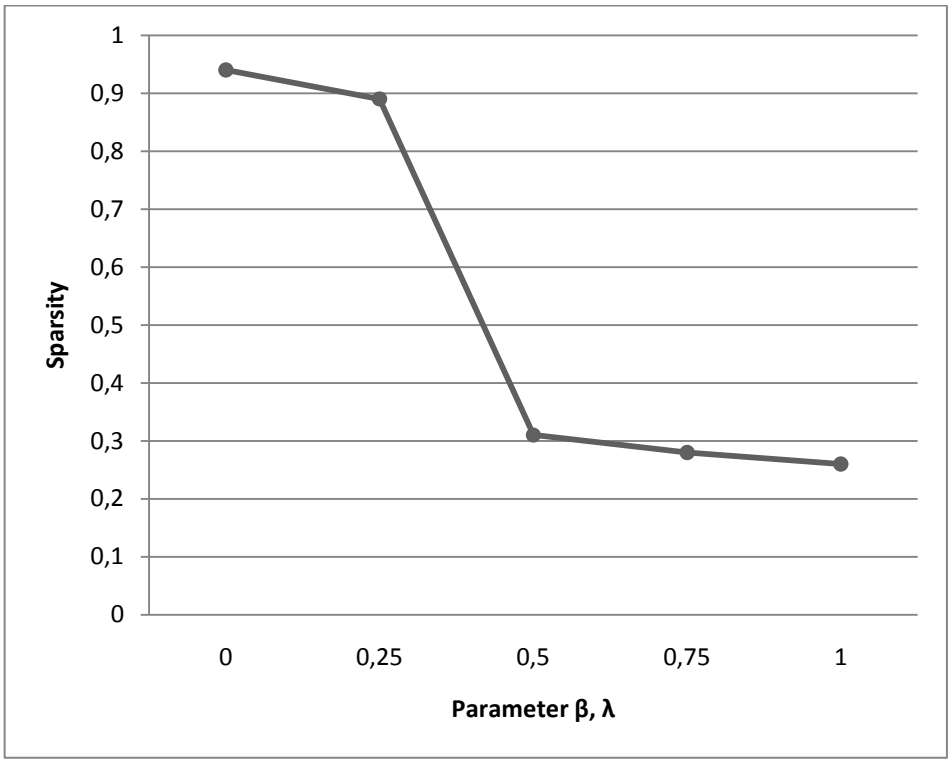

Figure 17. MovieLens200: Impact of β, λ Parameters on Sparsity

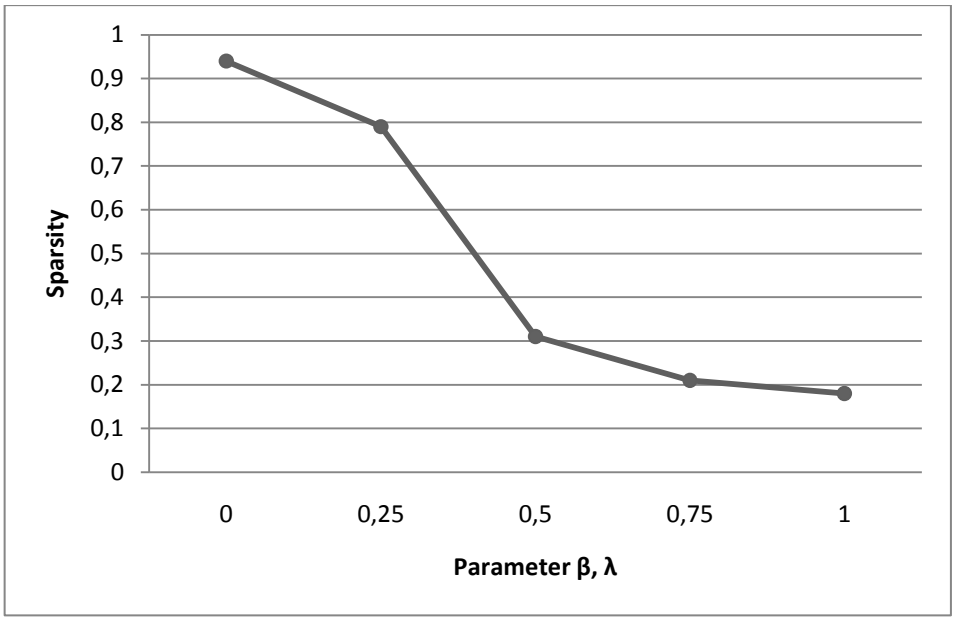

Figure 18. MovieLens100: Impact of β, λ Parameters on Sparsity

All figures reveal that values between 0.5 and 0.7 for β, λ parameters decreases sparsity in an effective way as seen in mean absolute errors of the β parameter.

CHAPTER 7

CONCLUSION

This thesis presents MoresysGOAL which is an ASP.NET web-based movie recommendation system, is written in c# programming language in Visual Studio 2010 environment. The main aim of the system is to improve sparsity problem successfully. In other words, this thesis tries to increase the density while aiming to maximize accuracy at the same time. Effective missing prediction and content-boosted collaborative filtering approaches throw light on how to deal with deal with this sparsity problem. In addition, the contents of movies each of which is taken from IMDb by using information extraction methods are also play a crucial role for calculating the item similarity. Both of these content information and item-based collaborative filtering method are then used together in prediction process.

In the beginning, current recommendation systems and main theoretical issues behind them are generally introduced. Afterwards several types of recommendation systems are experimented in so many data sets from movies to books. Subsequently, these systems are examined in both positive and negative directions provided by their applications. In the most crucial part, comprehensive amount of study is done about overall system design and the prediction approach. Finally, MoresysGOAL is compared with other successful literature studies in similar experimental setups. In other words, parameters used in evaluation, are set to the most suitable values when testing process is done between MoresysGOAL and other accomplished systems.

It is revealed in experimental setup that MoresysGOAL gives better accuracy results than traditional user-based and item-based collaborative filtering methods at most conditions. MoresysGOAL uses different content weights from GOCBCF approach; these content weights improve item similarity calculations. Moreover, Mahout implementations of collaborative filtering algorithms contribute to this improvement in a positive way. Further, overall prediction process is supported by EMDP resulting in increased density and maximum accuracy.

In the near future, MoresysGOAL will be installed in IIS Server and so it will be published in internet. Datasets of MoresysGOAL will be updated continuously and it

will make online actual rating predictions to the users whose habits are changing day by day. As a result, MoresysGOAL can be sensitively satisfying current user tastes.

As soon as it will be attempted that different types of user-based and item-based collaborative filtering approaches will be put in prediction phase of MoresysGOAL application. Hence, much more successful results might be obtained. The Prediction approach behind MoresysGOAL can also be tried in different datasets to test harmony performance of MoresysGOAL system. Further, Mahout Hadoop will be used for large scale datasets of GroupLens research group to overcome scalability problems of recommendation systems.

Managers of production companies or publishers are also users who have the predicted list of movies from the most appropriate to the incompatible ones. Before they make an investment on a movie, they may want to consider which kind of genre will increase the movie rating. Also, from the cast list they might find the most convenient pair of actors and actresses to provide an improvement for rating value. Moreover, directors may prefer most suitable cast and content information based on the result list of MoresysGOAL prediction approach.

REFERENCES

Adomavicious, G., Tuzhilin, A. (2005). Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. *IEEE Trans. Knowl. Data Eng.,* pp. 734-749.

Ahn, S., Shi, C. K. (2008). Exploring Movie Recommendation System Using Cultural Metadata. *International Conf. on Cyberworlds.*

Alag, S. (2008). Collective Intelligence in Action, pp. 349-387.

Amazon.com, http://www.amazon.com, accessed July 2, 2012

Armstrong, J.S. (2001). Principles of Forecasting – A Handbook for Researchers and Practitioners. Kluwer Academic.

Balabanovic, M., and Shoham, Y. (2007). Fab: Content-based, Collaborative Recommendation. Comm. ACM, vol. 40, no. 3, pp. 66-72.

Banko, M., Cafarella, M. J., Soderland, S., Broadhead, M., and Etzioni, O. (2007). Open information extraction from the web in IJCAI, pp. 2670–2676.

Basu, C., Hirsh, H., Cohen, W. (1998). Recommendation as Classification: Using Social and Content-Based Information in Recommendation", In: AAAI/IAAIAAAI Press / The MIT Press, pp. 714-720.

Bennett, J., and Lanning, S. (2007). The Netflix Prize, In Proceedings of KDD Cup and Workshop, San Jose, CA, USA.

Burke, R. (2007). Hybrid web recommender systems, The Adaptive Web: Methods and Strategies of Web Personalization, volume 4321 of Lecture Notes in Computer Science. Springer-Verlag, Berlin Heidelberg New York.

Burke, R. (2000). Knowledge-based Recommender Systems, In: A. Kent (ed.): Encyclopedia of Library and Information Systems, 69, Sup. 32.

Debnath, S., Ganguly, N., Mitra, P. (2008). Feature Weighting in Content Based Recommendation System Using Social Network Analysis. *Proceedings of the 17th international conference on World Wide Web*, pp. 1041-1042, ACM New York, NY.

EachMovie, http://research.compaq.com/SRC/eachmovie, accessed July 2, 2012

Fayyad, U. M., .J., Piatetsky-Shapiro, G., Smyth, P. (1996). From Data Mining to Knowledge Discovery: An Overview", in: Advances in Knowledge Discovery and Data Mining, AAAI Press, Menlo Park, pp. 1 – 34.

Fleder, D. M. and Hosanagar, K. (2007). Recommender systems and their impact on sales diversity. *Proceedings of the 8th ACM conference on Electronic Commerce*, pp. 192-199.

Ghazanfar, M., RrügelBennett, A. (2010). An Improved Switching Hybrid Recommender System Using Naïve Bayes Classifier and Collaborative Filtering.

Goldberg, D., Nichols, D., Oki, B. M., and Terry, D. (1992). Using collaborative filtering to weave an information tapestry. *Communications of ACM*, vol. 35, no. 12, pp. 61–70.

Goldberg, K., Roeger, T., Gupta, D., and Perkins, C. (2001). Eigen-taste: a constant time collaborative filtering algorithm. *Information Retrieval*, vol. 4, no. 2, pp. 133-151.

Good, N., Schafer, J. B., Konstan, J. A., Borchers, A., Sarwar, B., Herlocker, J., and Riedl, J. (1999). Combining collaborative filtering with personal agents for better recommendations. *In Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI-99)*, pp. 439–446.

Grooveshark, http://www.grooveshark.com, accessed July 2, 2012

Gunawardana, A., Meek, C. (2009). A Unified Approach to Building Hybrid Recommender Systems. *Proceedings of the third ACM conference on Recommender systems*, Pages 117-124, ACM New York, NY, USA.

Han, J., Kamber, M. (2006). Data Mining: Concepts and Techniques, 2nd ed. *The Morgan Kaufmann Series in Data Management Systems*, Jim Gray, Series Editor Morgan Kaufmann Publishers.

Herlocker, J., Konstan, J., Borchers, A., and 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*, pp. 230–237.

Hill, W., Stead, L., Rosenstein, M., and Furnas, G. (1995). Recommending and Evaluating Choices in a Virtual Community of Use. *Proc. Conf. Human Factors in Computing Systems*.

Hofmann, T., and Puzicha, J. (1999). Latent Class Models for Collaborative Filtering. *In Proceedings of the 16th International Joint Conference on Artificial Intelligence*.

IMDbPY, http://www.imdbpy.sourceforge.net, accessed July 2, 2012

Internet Movie Database, http://www.imdb.com, accessed July 2, 2012

Krulwich, B. (1997). Lifestyle Finder: Intelligent User Profiling Using Large-Scale Demographic Data", *AI Magazine 18 (2)*, pp. 37-45.

Last.fm, http://www.last.fm, accessed July 2, 2012

Last.fm API, http://www.last.fm/api, accessed July 2, 2012

Linden, G., Smith, B., York, J. (2003). Amazon.com Recommendations Item-to-Item Collaborative Filtering.

Lim, Y. J, Teh, Y. W. (2007). Variational Bayesian Approach to Movie Rating Prediction. *KDDCup. 2007*, San Jose, California, USA.

Luo, H., Niu, C., Shen, R., Ulrich, C. (2008). A collaborative filtering framework based on both local user similarity and global user similarity.

Ma, H., King, I., and Lyu, M. R. (2007). Effective missing data prediction for collaborative filtering. *in Proc.* 30^{th} *annual international ACM SIGIR conference on research and development in information retrieval*, pp. 39-46, New York.

Mahmood, T., and Ricci, F. (2009). Improving Recommender Systems with Adaptive Conversational Strategies. *Proceedings of the 20th ACM conference on Hypertext and hypermedia*, pp. 73-82.

Marlin, B. (2004). Modeling user rating profiles for collaborative filtering. Advances in Neural Information Processing Systems, 16, pp. 627–634.

Marmanis, H., Babenko, D. (2009). Algorithms of the Intelligent Web. pp. 69-120.

Melville, P., Mooney, R., and Nagarajan, R. (2002). Content-boosted collaborative filtering for Improved Recommendations. *in. Proc. 18th Conference on Artificial Intelligence*, pp. 187-192, Edmonton, Canada.

Miller, B. N., Konstan, J. A., and Riedl, J. (2004). PocketLens: toward a personal recommender system," ACM Transactions on Information Systems, vol. 22, no. 3, pp. 437-476.

Mitchell, T. (1997). Machine Learning. New York, NY:McGraw-Hill.

Mobasher, B. (2007). Recommender Systems. Kunstliche Intelligenz, Special Issue on Web Mining. No. 3, PP. 41-43, BottcherIT Verlag, Bremen, Germany.

MovieLens Recommendations, http://movielens-umn.edu, accessed July 2, 2012

Mukherjee, R., Dutta, P. S., and Sen, S. (2001). MOVIES2GO - A new approach to online movie recommendation. *In Proceedings of the IJCAI Workshop on Intelligent Techniques for Web Personalization.*

Murthi, B.P.S, and Sarkar, S. (2003). The Role of the Management Sciences in Research on Personalization. *Management Science*, vol. 49, no. 10, pp. 1344-1362.

Nadi, S., Saraee, M., H., Bagheri, A. (2011). A Hybrid Recommender System for Dynamic Web Users. International Journal Multimedia and Image Processing (IJMIP), Volume 1, Issue 1.

O'Mahony, M., O'Brien, M., Boydell, O., and Smyth, B. (2008). A Recommender System Approach to Enhance Web Search and Query Formulation.

OPL Studio 6.3, http://www-01.ibm.com/software/integration/optimization/cplex-optimization-studio/, accessed July 2, 2012

Owen, S., Anil, R., Dunning, T., and Friedman, E. (2011). Mahout In Action. *Mannings Publications Co*., pp. 41-70.

Özbal, G., Karaman, H., and Alpaslan, F. N. (2011). A Content-Boosted Collaborative Filtering Approach for Movie Recommendation Based in Local and Global Similarity and Missing Data Prediction.

Pazzani, M. J., Billsus, D. (2007). Content-Based and Demographic Filtering. The Adaptive Web, pp. 325-341.

Personallogic, http://www.personallogic.com, accessed July 2, 2012

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., and Riedl, J. (1994). Grouplens: an open architecture for collaborative filtering of netnews. in Proceedings of the ACM Conference on Computer Supported Cooperative Work, pp. 175–186, New York, NY, USA.

Ricci, F., Rokach, L., and Shapira, B. (2011). Introduction to Recommender Systems Handbook.

Russel, S., Norvig, P. (2010). Artificial Intelligence A Modern Approach. Third Edition.

Salton, G. 1989. Automatic Text Processing. *Addison-Wesley*.

Sarker, R.A., Newton, C.S. (2008). Optimization Modeling: A Practical Approach. CRC Press, Taylor&Francis Group, pp. 3-14.

Sarwar, B., Karypis, G., Konstan, J., Riedl, J. (2001). Item-Based Collaborative Filtering Recommendation Algorithms", WWW10, Hong Kong.

Schafer, J.B., Konstan, J.A., and Reidl, J. (2001). E-Commerce Recommendation Applications. Data Mining and Knowledge Discovery, Kluwer Academic, pp. 115-153.

Shardanand, U., and Maes, P. (1995). Social Information Filtering: Algorithms for Automating 'Word of Mouth'," Proc. Conf. Human Factors in Computing Systems.

Sheth, B., and Maes, P. (1993). Evolving Agents for Personalized Information Filtering. *Proc. Ninth IEEE Conf. Artificial Intelligence for Applications*.

Spiegel, S., Kunegis, J., Li, F. (2009). Hydra: A Hybrid Recommender System. CNIKM'09, Hong Kong, China.

Su, X., Khoshgoftaar, T. M., "A Survey of Collaborative Filtering Techniques", 2009

Taha, H. A. (2007). Operations Research: An Introduction (8th Edition). Pearson Education, Inc., pp. 335-348.

Ungar, L. H., and Foster. D. P. (2007). Clustering Methods for Collaborative Filtering. *In Proc. Workshop on Recommendation Systems at the 15th National Conf. on Artificial Intelligence*. Menlo Park, CA: AAAI Press.

Wang, J., Vries, A., Reinders M. (2006). Unifying User-based and Item-based Collaborative Filtering Approaches by Similarity Fusion. *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, Pages 501 – 508, ACM New York, NY, USA.

Winston, W. L., Goldberg, J. B. (2003). Operations Research: Applications and Algorithms.

Xue, G., Lin, C., Yang Q., Xi, W., Zeng, H., Yu, Y., Chen, Z. (2005). Scalable Collaborative Filtering Using Cluster-based Smoothing. SIGIR'05, Salvador, Brazil.

Youtube, http://www.youtube.com, accessed July 2, 2012

Yu, K., Schwaighofer, A., Tresp, V., Xu, X., and Kriegel, H.-P. (2004). Probabilistic memory-based collaborative filtering. *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no.1, pp. 56–69.

Zang, Y., Callan, J., and Minka, T. (2002). Novelty and Redundancy Detection in Adaptive Filtering" Proc. 25th Ann. Int'l ACM SIGIR Conf., pp. 81-88.

Zhang, M. (2009). Enhancing diversity in top-n recommendation. *In: RecSys '09: Proceedings of the third ACM conference on Recommender systems*, pp. 397–400. ACM, New York, NY.

Ziegler, C.-N., McNee, S.M., Konstan, J.A., and Lausen, G. (2005). Improving Recommendation Lists Through Topic Diversification. *International World Wide Web Conference. Proceeding of the 14th international Conference on World Wide Web*.