Hybrid recommender system using association rules

Alex Cristache

A thesis submitted to
Auckland University of Technology
in partial fulfillment of the requirements for the degree of
Master of Computer and Information Sciences (MCIS)

2009

School of Computer and Mathematical Sciences

Primary Supervisor: Russel Pears

Table of Contents

Attestation of authorship

“I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person now material which to a substantial extent has been accepted for the qualification of any other degree or diploma of a university or other institution of higher learning, except where due acknowledgement is made in the acknowledgements.”

Yours sincerely,

(Alex Cristache)

Acknowledgements

Having moved from another country just when I started studying for this masters degree was a challenge for me. It involved adjusting to a new system of learning and the use of the English language every day. This dissertation represents the end point of my masters degree. A degree that helped me refine the undergraduate studies I took in my home country. It also gave me the opportunity to ease the transition to New Zealand by meeting people from all over the world that were going through the same situation as me.

Studying at AUT also gave me the opportunity to find a job in an area that interests me and that I still have now. I appreciate the support that my employer and colleagues gave me. I acknowledge the advices they gave me based on the experience they gained from their postgraduate studies.

To my family, I acknowledge the support you gave me throughout the years and the way you made me focus on my studies from an early age.

To all the lecturers I had at AUT, I acknowledge the way you made me find the parts that interested and motivated me in all the classes I took.

To my supervisor, Dr. Russel Pears, I acknowledge the time and advices you gave me. I thank you for the way you kept me on track and focused on finding solutions for the issues encountered along the way. I admire the devotion you have for your profession and the way you try and make sure your students have gained knowledge and experience from their years of studying.

To my supervisor, Dr. Dave Parry, I acknowledge the way you inspired me in seeing the broader picture of what recommender systems are.

Abstract

Recommender systems are increasingly being used in today's world. Collaborative filtering, together with association rules mining are probably the most widely used methods to implement recommender systems.

In this dissertation we undertake a review of past research conducted in the area of recommender systems with the focus being the use of association rule mining.

We propose a novel methodology that combines the use of association mining with the use of distance metrics such as the Jaccard measure to identify movies that belong to the same genre. Our experimental results on the MovieLens dataset shows that the use of the Jaccard metric improved the coverage of recommendations over the use of the standard association rule mining method.

Chapter 1: Introduction

The development of the Internet provided more ways for people to interact but also a place where they could find information about almost everything and anything. Recommender systems can be considered a way of combining these two aspects in order to help people find the information they need or something they would be interested in.

Recommender systems are used in various online applications from e-Commerce to search engines. There are a number of techniques used to implement recommender systems, each with its advantages and disadvantages. Hybrid systems intend to combine two or more of these techniques in order to obtain better results.

Collaborative filtering recommender systems are the most commonly used systems (Burke, 2002). They involve the use of the information provided by other users to make suggestions to a particular user. This can be compared to what happens in real life when an item is purchased based on the recommendation made by a friend. Collaborative filtering systems differ in the way they use the information provided by other users to link it to the information available about the user that it needs to make a prediction for. A type of collaborative filtering is the use of association rules.

Association rules find patterns in the information available about user preferences. These patterns are then used to make predictions based on the information available for the selected user.

The purpose of this dissertation is to propose a hybrid recommender system based on association rules. The system will be used to recommend movies. The initial idea was to combine association rules with a movie ontology. Due to the fact that such an ontology was not available and difficult to create, a different way to create a relationship between movies was needed. This was chosen to be a similarity coefficient of the genres the movies belongs to.

The first part of the dissertation will provide a literature review of the recommender systems area with an emphasis on association rules based systems. A classification of recommender systems will be discussed and some of the challenges in this field will be identified.

The next part of the dissertation concentrates on the hybrid system proposed. The methodology used to develop the system is constructive research. The framework used to develop the system is presented and discussed, along with a presentation of a novel algorithm that was developed as part of this research. The metrics used to assess the system were chosen as precision, recall and the F1 metric.

The system is intended to recommend movies. The dataset chosen is MovieLens and was taken from GroupLens (www.grouplens.org). Apart from user ratings of the movies, the dataset was chosen as it contains detailed information about users and movies. For example, user demographic data is present. For movies, the aspect that was used in the hybrid algorithm is the genre it belongs to.

Next the experimental setup is presented. Experiments were conducted in various states of the development of the hybrid system in order to asses what progress each alteration brought. The results of the experiments are discussed and compared with other work on the same dataset. Systems that have used the same dataset for evaluation are the ones presented by Sarwar et al. (2000) or Kim and Kim (2003).

The final chapter of the dissertation discusses if the system has fulfilled its purpose and looks at what future developments or improvements can be made.

Chapter 2: Related research

2.1 Introduction

Recommender systems development was driven by e-Commerce but there are also other applications for them such as search results and news portals customization. Various techniques have been used, including the nearest neighbor algorithm (Herlocker et al., 2004), association rule mining (Demiriz, 2004) and neural networks (Changchien and Lu, 2001), to name but a few. Hybrid techniques were implemented to overcome some of the deficiencies in the aforementioned techniques. The deficiencies include performance aspects, but also trust, security and privacy issues.

2.2 Applications of recommender systems

According to Resnick and Varian (1997) one of the first applications of recommender systems was in e-Commerce. Businesses saw an opportunity in presenting to their internet customers products they might be interested in. The rationale for this approach was the abundance of products that made user choices and navigation very difficult (Resnick and Varian, 1997). This generalizes to what Lam and Riedl (2004) call information overload.

Changchien and Lu (2001) also see recommender systems as ways to help internet shoppers select the products best suited for them. Another system that provides recommendations for e-Commerce applications is the one presented by Tran and Cohen (2000).

The results provided from recommender systems are not limited to suggesting what users can buy. Ansari et al.(2000) identify other applications for recommender systems. One of them is that of helping users search the internet. Search engines can provide customized results for each user. The customization also applies to other types of websites, such as news portals. The evolution to a semantic web was seen by Middleton et al.(2004) as a method to further improve the way users browse the internet. Also the results of the search engines will not be limited to the keyword queries entered. However, as Middleton et al.(2004) state, the semantic web is still in his infancy as it is dependant on the metadata that authors attach to their websites. The transition to the semantic web is something that will take a long time considering the amount of information that is available at the moment online and that needs to be transformed so that ontologies can be used to classify it. Middleton et al.(2004) also present another application of recommender systems, which is to do with finding similar research articles to a given article that has been selected by the user.

There are other recommender systems that are based on studying the browsing patterns of internet users. These are based only on collecting and analyzing the click stream data, a process that will be invisible to the user. Such systems are presented by Mobasher et al. (2000), Cho et al. (2002) and Fu et al. (2000). An advantage of such systems is that they are not subjective, based only on what the user has selected on their profile and, as Mobasher et al. (2000) suggests, these systems can adapt to the changes in user preferences. Fu et al. (2000) raise concerns about the privacy issues related to using the automatically collected weblogs to find information about users.

When recommender systems were still in their infancy, their use in customization of the learning process was tried. Linton et al. (1998) have proposed two approaches to help users learn how to use a software application. The two approaches are intelligent tips – tips that appear at the launch of the program and a Skillometer – where a user can compare the functions of the program he is using with what similar users are using.

The use of recommender systems is not limited to classical online applications. Lawrence et al. (2001) propose a system that works in supermarkets. A PDA is used by customers to select the products they want to buy and these are gathered and put on hold for them to pickup. The purpose of the recommender system is to present to users items they could be interested in without them visiting the supermarket. A similar system that can work on PDAs that are not always connected to the internet is *MovieLens Unplugged* that was presented by Miller, Albert, Lam, Konstan and Riedl (2003). The applications of recommender systems in mobile environments are further developed in the PocketLens system by Miller et al. (2004). These two systems recommend movies to users even if they are not connected to the main database.

2.3 Techniques used

According to Burke (2002) the most commonly used technique in recommender systems is collaborative filtering. Collaborative filtering is compared by Resnick and Varian (1997) with what happens in reality when people make decisions based on the opinions of others. The other techniques that Burke (2002) used to classify recommender systems are presented in Table 1.

Technique	Background	Input	Process
Collaborative	Ratings from U of items in I.	Ratings from **u** of items in I.	Identify users in U similar to **u**, and extrapolate from their ratings of **i**.
Content-based	Features of items in I	**u**'s ratings of items in I	Generate a classifier that fits **u**'s rating behavior and use it on **i**.
Demographic	Demographic information about U and their ratings of items in I.	Demographic information about **u**.	Identify users that are demographically similar to **u**, and extrapolate from their ratings of **i**.
Utility-based	Features of items in I.	A utility function over items in I that describes **u**'s preferences.	Apply the function to the items and determine **i**'s rank.
Knowledge-based	Features of items in I. Knowledge of how these items meet a user's needs.	A description of **u**'s needs or interests.	Infer a match between **i** and **u**'s need.

Table 1. Techniques used in recommender systems (Burke, 2002)

The classification is based on data that is collected automatically (background) and the data that is introduced by the users (input).

Collaborative filtering techniques are based on the ratings that users gave to the products. These ratings are used to find similar users and based on that community of users, products (movies or books) are recommended.

The demographic approach also finds similar users. The difference is that previous ratings or transactions are not used. Instead the characteristics of the users obtained through a questionnaire are used to group them.

Content-based techniques are used to filter data based on a user profile. The user profile is built by finding habits of the users in the data available. An example of such a process is the one presented by Changchien and Lu (2001) where first a filtering is done using neural networks.

Utility-based and knowledge-based systems are similar as they are not based on previous transactions but on user needs. Other factors apart from item characteristics are taken into account. For example, as Burke (2002) presents the availability of an item is matched with how soon a user needs the item. Both utility-based and knowledge-based systems create a user profile that reflects the needs of the user. In the case of utility systems, there is a function that calculates the utility of a recommendation for the user.

From the high level classification presented by Burke (2002) a closer look can be taken on the actual techniques used in the implementation of recommender systems. One of these techniques is the one used to develop the hybrid system presented in this dissertation: association rules. It can be classified under the collaborative filtering category as it is building its recommendations using the information provided by other users.

The definition of association rules is given by Agrawal (1993). Association rules were originally used in market basket analysis. An example of a rule can be: 70% of users that bought milk also bought beer. Another way of expressing this is: milk -> beer with a confidence of 0.7. There are two main parameters that are involved in the building of association rules: support and confidence. Support is connected to the coverage that a rule has while confidence is related to the trust that can be put in the prediction that the rule makes. When using association rules in recommender systems, thresholds for support and confidence are set. Another option is to set a number of rules to be generated as it is implemented in their system by Lin et al. (2002).

Support and confidence can also be used to rank the recommendations made. In the case of the system presented in this dissertation only the confidence is taken into account. For each item that is recommended, the confidence of all the rules that recommended that item will be added. Based on these sums the recommendations are ranked. Demiriz (2004) use the confidence of the rules but this is multiplied with a similarity coefficient. The similarity coefficient is calculated between the items involved in a rule and the items from the user's profile.

The association rules can be created as item associations or as user associations. In the case of item associations, the items liked by a user can be considered a transaction. There will be a number of transactions equal to the number of users. The rules that will be created will be in the form: when item X and item Y are rated together, also item Z is rated. In the case of user associations, the number of transactions will be equal to the number of movies. Each transaction will contain the users that have rated that movie. The resulting rules are in the form: when user X has rated movie M, user Y has also rated that same movie M. Both approaches were tried in the development of the hybrid algorithm proposed in this dissertation. Also the two approaches used by Lin et al. (2002) and also by Demiriz (2004) were also experimented with.

Association rules can be used on databases that contain user ratings for items (Demiriz, 2004) but also on web logs. In this case, a complex preprocessing step is required. This is described by Mobasher et al. (2000). The preprocessing will perform data cleaning, identify the user and the sessions, but also the path the user followed. From the identification of the pages viewed by the user the items she has viewed can be found. This can be used as an input for the association rules building process. Cho et al. (2002) combine the association rules found from the web logs with a decision tree to select appropriate users for the recommendations. Also a taxonomy is used. Products are classified in classes. Instead of recommending a product, the initial recommendation process will recommend a class of products. From this class the most bought products or the products with the best click to buy ratio will be recommended to the user. The hybrid system proposed in this research will present to the user similar items to the ones resulted from using association rules.

In some cases, collaborative filtering is not sufficient. Hybrid systems are developed that also include item attributes in the selection of the recommendations. Such a system is the one presented by Ansari et al.(2000). Their system is using a Bayesian network approach that combines collaborative filtering with content filtering. A Bayesian approach is also used in the system proposed by Schein et al. (2002) that recommends movies from the MovieLens dataset taking into account information such as the actors playing in each movie. Another hybrid system is the one proposed by Tran and Cohen (2000). The system combines a collaborative filtering technique with a knowledge based one. The advantages are that it can give personalized recommendations without having the need of a large database of previous transactions.

Another hybrid system based on association rules is the one presented by Changchien and Lu (2001). The association rules are not applied to the entire database. Clusters are first built using a neural network. The association rules will also find the relationships between the clusters.

2.4 Challenges

Recommender systems are usually measured on how accurate they are. Amongst the metrics that asses the accuracy of a recommender system, Herlocker et al. (2004) identify precision and recall. Precision is defined as the ratio of items predicted correctly to the total number of items predicted. Recall is defined as the ratio of items predicted correctly to the total number of items that can be selected. In other words, precision is defined as the probability that a recommended item is relevant while recall is the probability that a relevant item is recommended (Herlocker et al., 2004).

Herlocker et al. (2004) find that the accuracy metrics do not necessarily asses a recommender system's ability to make suggestions that are needed by the user. This view is also shared by McNee et al. (2006). For example, in the case of a virtual book store, a recommender system can identify that a user likes a particular writer and recommend all the books available from that author. This prediction might have a very good accuracy but it lacks in other aspects. McNee et al. (2006) identify these aspects as being similarity and serendipity. Similarity is related to the book example. There needs to be a diversity in the type of items that the system recommends. Serendipity refers to the user receiving unexpected recommendations. This is very difficult to asses without user feedback. Usually collaborative recommender systems suggest items that are popular. This might generate a good accuracy, but might not be useful for what the user is looking at a particular moment. Another aspect that needs to be taken into account when analyzing a recommender system is considered by Herlocker et al. (2004) to be coverage. A system that covers more items is more likely to produce more diverse recommendations.

Classic collaborative filtering methods find similar users to build their recommendations on. Massa and Avesani (2004) propose a system that also takes into account the trust between users. Recommendations are made based on the trust between users rather than the similarity between the items that the users have rated. A similar approach is made by O'Donovan and Smyth (2005).

As the Internet and the computing devices are becoming more and more ubiquitous, another issue that should be taken into account for the future of recommendation systems is that portability. Two main challenges are identified by Miller et al. (2004) when it comes to portability: privacy and offline use. A portable recommender system needs to be able to protect the data it holds in an open and unsecured environment. Also the computations that the system is making need to be able to run on a portable device, such as a PDA. Often the device will not be connected to a network so a solution for offline recommendations needs to be found.

Chapter 3: Proposed methodology

3.1 Introduction

This chapter describes the reasoning that governed the building of the hybrid recommendation algorithm that is being proposed. The evolution of the algorithm over a number of develop-test cycles is described. We also present the measurement parameters used. A short review of research methodologies used in the area of data mining is also presented in this chapter in order to justify the choice that was eventually used in the development of the hybrid algorithm.

3.2 Research methodology

There are several research methods that are used in the computer science field; these can be classified into two major approaches. According to Spens and Kovacs (2006) these two main approaches are the deductive and inductive approaches. Deductive research can be considered to be the main form of research. It generally consists of two phases. The first one is a theoretical phase in which new hypotheses are generated. The second phase is an empirical one and is concerned with testing the hypotheses that were generated in the first phase. On the other hand, inductive research starts with empirical observations and based on them try to build rules that are supported by some metric that is computed from the underlying data. Thus in the deductive approach a given theory is tested, whereas in the inductive approach a new theory is constructed.

Spens and Kovacs (2006) also identify a third new approach called abductive. This approach consists in finding amongst a theory that already exists, a combination that will explain a particular phenomenon. It starts like the inductive approach from the empirical aspect but it also takes into account previous research. Another characteristic of this approach is the iterative nature of the process of finding the rule that will explain the phenomenon in discussion. Action research and constructive research are two research methods that according to Spens and Kovacs (2006) have a framework that fits the abductive approach.

A closer look on the research approaches used in the area of data mining is presented in Pechenizkiy et al. (2005). Four main frameworks are identified. The first one is a database perspective that introduces the notion of inductive databases. An inductive database is created by selecting information that follows a given logic from the main database. New knowledge can then be generated by querying the new database.

The second framework is from a statistical point of view. All data mining tasks can be considered statistical tasks applied to larger datasets (reductionist approach) or tasks that try to find connections between the distribution of the variables (probabilistic approach).

A third approach is that of considering that all data mining techniques are used to compress data by creating a structure that can be used to describe it. Methods such as association rules, decision trees or clustering can be considered as ways to compress some of the data.

The final approach presented by Pechenizkiy et al. (2005) is that of constructive induction. This approach is similar to the induction approach described in Spens and Kovacs (2006). There are two phases, one that scans the data and a second one that tries to find the best hypotheses for it. The constructive aspect comes into play by trying to expand the space that is being analyzed. There is also a destructive variant that tries to shrink the representational space by doing feature selection.

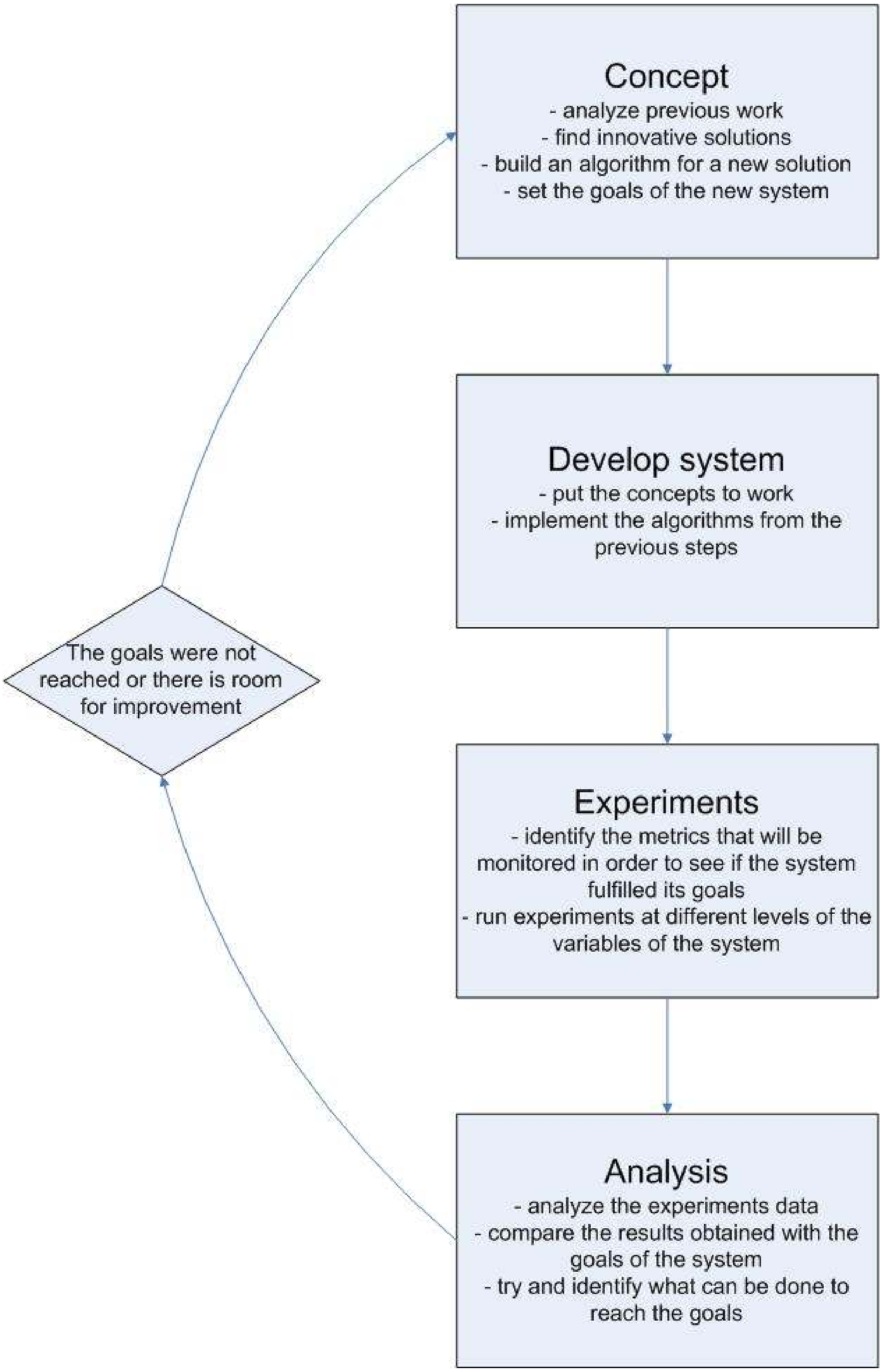

Figure 1. Research methodology framework

The research method used in order to develop the hybrid recommender system is constructive research. It can be considered an abductive approach as it starts by using existing recommendation methods on a particular dataset and tries to find a combined solution that will best fit the dataset considered. Also there are a few iterations that led to the final system that are presented in the diagram in Figure 1. Each of the iterations consisted of four steps. The first step was conceptual, whereby existing work in the area was used as a guide to finding new solutions that could be adapted to work in a recommender system. In this step the goals of the system and the hypothesis are set. The next step handles the implementation of the solutions that were proposed in the first step. It is at this stage that actual data issues will be handled. The implementation is then used to experiment and find if the goals of the system are met. It is important to define the parameters that are more influential and also what needs to be recorded in order to perform the analysis. The next step in the process is the analysis that will be critical for driving the development of the system. Based on the results of the analysis after each of the iterations a new version of the system was proposed.

3.3 Hypothesis

The purpose of each piece of research is, in general, the generation of new knowledge. This can be achieved by either finding a completely innovative idea or by looking at previous work in order to find solutions that apply to a particular problem. In either case, the purpose of the research needs to be defined. It can start as a general goal or as a specific research question but the scope of the research needs to be stated clearly in the form of a hypothesis. The hypothesis will either then be confirmed or rejected following the experimentation phase.

Based on the initial literature research process two main hypothesis were defined.

Hypothesis 1. Association rules are not powerful enough to be used in the production of a good performance recommender system.

The implementations that used association rules that were found in the initial literature review did not perform well, especially on a sparse dataset. The experiments carried out in this dissertation will try to prove if that is the case for the dataset chosen.

Hypothesis 2. Using genre information can improve the performance of a movie recommender system based on association rules.

A way to improve the performance of an association rules based recommender system needs to be found if the first hypothesis is proved to be correct. Various methods were found in literature and their performance is discussed and compared to the one used in the proposed system. In general, the alternative approaches try to apply already existing technologies such as nearest neighborhood techniques along with association rules that employ different methods for generating the association rules. The hybrid system proposed will enrich the items recommended by the classical association rules approach with similar movies based on their genre. How the performance is affected by this will be analyzed in order to find if the hypothesis is true or false.

3.4 Performance metrics

In order to see how the proposed system performed after each of the iterations a performance measure needs to be defined. For recommender systems two aspects are important in general, response time and the quality of the recommendation process. Because the nature of the system allowed it to be run offline the response time aspects were considered to be outside the scope of this study. Most of the computations can be done offline so this aspect will not influence the time needed to present the recommendations to the users.

The remaining aspect concerning the performance of the system is that of the quality of the recommendations made. Vozalis and Margaritis (2003) present an in-depth analysis of the measurements that are used for recommender systems. These include accuracy measurements, like MAE (Mean Absolute Error), coverage and recall/precision related measures.

In most of the papers found in literature, recall and precision are used. These are calculated for each user. Due to the fact that precision can often be improved by sacrificing recall and vice versa, a measure based on both was created. This measure is the F1 metric.

$$precision = \frac{\text{number of relevant hits}}{\text{number of recommendations}}$$

$$recall = \frac{\text{number of relevant hits}}{\text{number of ratings in test set}}$$

$$F1 = \frac{2 * \text{precision} * \text{recall}}{\text{precision} + \text{recall}}$$

Precision records the accuracy of the recommendations provided, while recall tests the extent of their coverage. Because the dataset used is very sparse the main objective in developing the hybrid algorithm was to improve recall. Precision and the F1 metric value were recorded for the experiments in spite of the fact that were low due to sparsity of the dataset used. The low precision and F1 values was also noted by Kim and Kim (2003) that developed a recommender system on the same dataset that was used as a test-bed for the hybrid system presented in this dissertation. Kim and Kim (2003) also use recall as their performance metric and so comparisons with the hybrid system that we developed was possible.

Two subsets are used in testing the system: train and test. The relevance of the hits is determined based on what each user has rated in the test dataset. The system recommends a movie based on the ratings of the user in the training subset and these recommendations are then compared to what the user has rated in the test subset. This can be considered a limitation but it is one imposed by the data available. A better option would be to obtain feedback on the recommendations made straight from the users as there might be movies they like but that are not present in the test subset.

3.5 Dataset description

The dataset used contains information about movies and users' ratings of those movies. It was taken from the GroupLens (*www.grouplens.org*) and it is named MovieLens. The dataset contains 100,000 of ratings from 943 users. The total number of movies is 1682.

There are several files that comprise the MovieLens dataset. These contain not only the actual ratings, but also information about the users and the movies. The ratings are presented in the u.data file and range from 1 to 5. Along with the actual rating, a timestamp is presented.

User ID	Movie ID	Rating value	Timestamp

Table 2. U.data file structure

User information is not limited to ratings only. The age, gender, occupation and zip code is available for all users. Each user has rated at least 20 movies. The user information is presented in the u.user file with a structure given in Table 3.

User ID	Age	Gender	Occupation	Zip code

Table 3. U.user file structure

Movie information contains the movie name, movie release date, IMDB link and a vector of 19 bits each corresponding to a genre. A movie can belong to more than a genre. When a movie belongs to a genre, the value corresponding to that genre is 1, otherwise it is 0. The file that contains the movie information is u.item and its structure is presented in Table 4.

Movie ID	Movie title	Release date	Video release date	IMDB url	Unknown	Genre vector

Table 4. U.item file structure

The genre vector is a binary vector with nineteen values. If the value is 1 then the movie can be classified in that corresponding genre. The nineteen genres are Action, Adventure, Animation, Children's, Comedy, Crime, Documentary, Drama, Fantasy, Film-Noir, Horror, Musical, Mystery, Romance, Sci-Fi, Thriller, War and Western. Figure 2 presents a screenshot of the u.item file.

```
1|Toy Story (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Toy%20Story%20(1995)|0|0|0|1|1|1|0|0|0|0|0|0|0|0|0|0|0|0|0
2|GoldenEye (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?GoldenEye%20(1995)|0|1|1|0|0|0|0|0|0|0|0|0|0|0|0|0|1|0|0
3|Four Rooms (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Four%20Rooms%20(1995)|0|0|0|0|0|0|0|0|0|0|0|0|0|0|0|0|1|0|
4|Get Shorty (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Get%20Shorty%20(1995)|0|1|0|0|0|1|0|0|1|0|0|0|0|0|0|0|0|0|
5|Copycat (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Copycat%20(1995)|0|0|0|0|0|0|1|0|1|0|0|0|0|0|0|0|1|0|0
6|Shanghai Triad (Yao a yao yao dao waipo qiao) (1995)|01-Jan-1995||http://us.imdb.com/Title?Yao+a+yao+yao+dao+waipo+qiao+(
7|Twelve Monkeys (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Twelve%20Monkeys%20(1995)|0|0|0|0|0|0|0|0|1|0|0|0|0|
8|Babe (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Babe%20(1995)|0|0|0|0|1|1|0|0|1|0|0|0|0|0|0|0|0|0|0
9|Dead Man Walking (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Dead%20Man%20Walking%20(1995)|0|0|0|0|0|0|0|0|1|0|0|
10|Richard III (1995)|22-Jan-1996||http://us.imdb.com/M/title-exact?Richard%20III%20(1995)|0|0|0|0|0|0|0|0|1|0|0|0|0|0|0|
11|Seven (Se7en) (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Se7en%20(1995)|0|0|0|0|0|0|1|0|0|0|0|0|0|0|0|0|1|0|0
12|Usual Suspects, The (1995)|14-Aug-1995||http://us.imdb.com/M/title-exact?Usual%20Suspects,%20The%20(1995)|0|0|0|0|0|0|1|
13|Mighty Aphrodite (1995)|30-Oct-1995||http://us.imdb.com/M/title-exact?Mighty%20Aphrodite%20(1995)|0|0|0|0|0|1|0|0|0|0|0|
14|Postino, Il (1994)|01-Jan-1994||http://us.imdb.com/M/title-exact?Postino,%20Il%20(1994)|0|0|0|0|0|0|0|0|1|0|0|0|0|0|1|0|
15|Mr. Holland's Opus (1995)|29-Jan-1996||http://us.imdb.com/M/title-exact?Mr.%20Holland's%20Opus%20(1995)|0|0|0|0|0|0|0|0|1
16|French Twist (Gazon maudit) (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Gazon%20maudit%20(1995)|0|0|0|0|0|1|0|0|
17|From Dusk Till Dawn (1996)|05-Feb-1996||http://us.imdb.com/M/title-exact?From%20Dusk%20Till%20Dawn%20(1996)|0|1|0|0|0|1|1
18|White Balloon, The (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Badkonake%20Sefid%20(1995)|0|0|0|0|0|0|0|0|1|0|0|
19|Antonia's Line (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Antonia%20(1995)|0|0|0|0|0|0|0|0|1|0|0|0|0|0|0|0|0|0|
20|Angels and Insects (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Angels%20and%20Insects%20(1995)|0|0|0|0|0|0|0|0|1
21|Muppet Treasure Island (1996)|16-Feb-1996||http://us.imdb.com/M/title-exact?Muppet%20Treasure%20Island%20(1996)|0|1|1|0|
22|Braveheart (1995)|16-Feb-1996||http://us.imdb.com/M/title-exact?Braveheart%20(1995)|0|1|0|0|0|0|0|1|0|0|0|0|0|0|0|1|
23|Taxi Driver (1976)|16-Feb-1996||http://us.imdb.com/M/title-exact?Taxi%20Driver%20(1976)|0|0|0|0|0|0|0|1|0|0|0|0|0|0|0|1
24|Rumble in the Bronx (1995)|23-Feb-1996||http://us.imdb.com/M/title-exact?Hong%20Faan%20Kui%20(1995)|0|1|1|0|0|0|1|0|0|0|
25|Birdcage, The (1996)|08-Mar-1996||http://us.imdb.com/M/title-exact?Birdcage,%20The%20(1996)|0|0|0|0|0|1|0|0|0|0|0|0|0|
26|Brothers McMullen, The (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Brothers%20McMullen,%20The%20(1995)|0|0|0|0|
27|Bad Boys (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Bad%20Boys%20(1995)|0|1|0|0|0|0|0|0|0|0|0|0|0|0|0|0|0|0|0
28|Apollo 13 (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Apollo%2013%20(1995)|0|1|0|0|0|0|0|1|0|0|0|0|0|0|1|0|
29|Batman Forever (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Batman%20Forever%20(1995)|0|1|1|0|0|1|1|0|0|0|0|0|0|
30|Belle de jour (1967)|01-Jan-1967||http://us.imdb.com/M/title-exact?Belle%20de%20jour%20(1967)|0|0|0|0|0|0|0|0|1|0|0|0|0|
```

Figure 2. U.item file screenshot

The dataset is divided into five subsets (u1 to u5). Each subset was split into one training set and one test set. This made things easier for a five cross fold validation experimentation. The training and test subsets are split in the ratio 80% to 20%, and are disjoint. These five subsets were used for the experiments.

The ratings were given in a Likert scale from 1-5, with 1 meaning that the movie was not liked by the user and 5 being the best rating. Only the ratings that rated movies positively (4 or 5) were taken into account.

3.6 Association rules definition and example

Agrawal (1993) give a formal statement of association rule mining for transaction databases. Let I = {i1, i2, . . . , im} be the universe of items. A set X of items is called an itemset. A transaction t = (tid ,X) is a tuple where tid is a unique transaction ID and X is an itemset. A transaction database D is a set of transactions. The count of an itemset X in D, denoted by count(X), is the number of transactions in D containing X. The support of an itemset X in D, denoted by supp(X), is the proportion of transactions in D that contain X. The rule X ->Y holds in the transaction set D with confidence c where c = conf(X->Y) and conf(X->Y) = supp(XY)/supp(X), where supp(XY) denotes the support of items X and Y occurring together.

In association rule mining the objective is to retrieve all rules of the form X->Y where supp(XY) > s and conf(X->Y) >c, with s and c being user-supplied thresholds on minimum support and minimum confidence respectively. (Agrawal, 1993).

In the case of the MovieLens dataset the association rules are built by finding the frequent itemsets that have a support greater than a user-supplied threshold. Based on these, the rules that have a given minimum confidence are selected.

The items are represented by the movie ID. A transaction is considered the itemset that contains all the movies rated by a user. This means that the transactional database will be formed from a number of transactions equal to the number of users.

For example, movies 10, 50 and 70 are rated together with a support of s = 0.5 (they occur jointly in half of the transactions). Also in a quarter (0.25) of the cases when movies 10 and 50 are rated together movie 70 is also rated. Then an association rule “movie10 and movie50 -> movie70” can be created. The support of the rule will be s = 0.5 while the confidence will be c = 0.25.

3.7 Hybrid recommender system

The initial idea was to build a neighborhood for each user and to use the association rules that were defined on the neighborhood to recommend new items for the selected user. The neighborhood could be created using the Euclidian distance based on user demographics such as the age, gender and occupation of the users. Another way to build the neighborhood was to introduce a similarity coefficient that measures the similarity between a given pair of users based on their ratings.

A matrix that represented users as rows and preferences (or ratings) for each of the items as columns was used to calculate this similarity coefficient. This was calculated by the cosine of the angle subtended by the vectors that represent the user preferences for each item extracted from the ratings matrix. These vectors are basically the rows in the matrix, each one corresponding to a different user. Each column in the matrix corresponds to a different movie.

The formula for the similarity (cosine) between vectors A = (x_1, x_2, …., x_n) and B = (y_1, y_2, …., y_n) is

$$Sim(A,B) = \cos ine(A,B) = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2}\sqrt{\sum_{i=1}^{n} y_i^2}}$$

The value 0 indicates that the vectors are far apart, having no term in common. Value 1 indicates that the vectors are identical. For example, for vectors A = (1, 0, 0, 1) and B = (0, 1, 1, 0) the cosine will be cosine(A, B) = (0+0+0+0)/sqrt((1+0+0+1)(0+1+1+0))=0. If C is considered to be identical to A, C = (1, 0, 0, 1) then cosine (A, C) = (1+0+0+1)/sqrt((1+0+0+1)(1+0+0+1)) = 2/sqrt(4) = 2/2 = 1.

The similarity coefficient obtained this way could be used as an alternative to the Euclidian distance when building a neighborhood.

Unfortunately, the neighborhood approach had to be abandoned. The issue was that building the association rules is a process that takes a lot of time. If the neighborhood approach was kept, it meant that for each user a neighborhood was built and then on this association rules were created. Because there are more than 900 users, this meant that results could not be presented to the user in a timely manner. The association rules are to be created using all the ratings. This allows for them to be created offline.

The next issue encountered was that only a very small number of movies were recommended for most users. A solution to overcome this was to build association rules on the inverted ratings matrix, i.e. with ratings as rows and users as columns. This has the effect of producing association rules with users (rather than movies) in the rule terms. After experimentation this solution was dropped as well as it proved to be very inefficient.

Another approach that was used to increase the coverage was to reduce the support and the confidence of the rules used. The influences of different levels of these parameters were tested.

There was still a lack in the quantum of movie recommendations for many users. In order to increase the coverage obtained from the algorithm the idea of introducing a movie ontology was considered. This proved to be a challenge as even if an existing otology for movies was used, it would be hard to classify the movies in the dataset with the information provided. However, because the dataset provided information about the movie genre and a movie generally belonged to more than one genre, relationships between movies could be found. The genres are represented as a binary vector. Using the Jaccard coefficient the similarity between movies can be obtained. Unlike the cosine measure of similarity between vectors the Jaccard measure was specially designed to measure the degree of measure between binary vectors as cited by Tan et al. (2006) This led to the final version of the algorithm: after the movies to be recommended based on the association rules are selected, an additional set of similar movies based on the Jaccard coefficient of the genre vectors were----- added to the list of movies recommended.

The value of the Jaccard coefficient is obtained with the formula

$$Jaccard = \frac{M_{11}}{M_{01} + M_{10} + M_{11}}$$

where M_{11} represents the number of instances where both vectors have the value 1 for the same column, M_{01} where only the second vector has value 1 for the same column and M_{10} when only the first vector has value 1 for the same column. The value 1 signifies movies that belong to the same genre, while 0 means that they come from different genres.

For example, for vectors A = (1, 0, 0, 1, 1) and B = (1, 1, 0, 0, 1), M_{11} = 2 (positions 1 and 5 in the vectors both have value 1), M_{01} = 1 (position 2 in vector A is 0 and 1 in vector B), M_{10} = 1 (position 4 is 0 in vector B and 1 in vector A). The Jaccard coefficient will be J = 2 / (1 + 1 + 2) = 2/4 = 0.5

A top-N cap was also implemented and the similar genre movies were found based just on these N movies for each user. The top-N movies were calculated using a matrix with items as rows and columns as users. Each time a rule recommends a movie to a user, the confidence of that rule is added to the cell in the matrix that corresponds to that movie and that user. After all the rules are parsed, the top-N movies for each user based on this index is chosen.

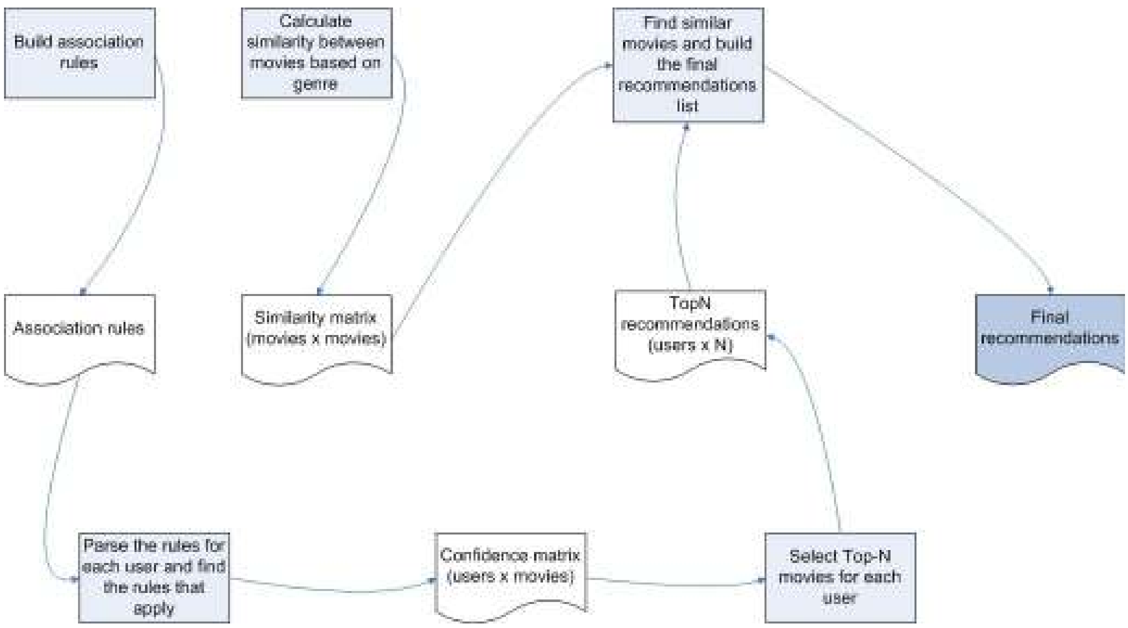

Figure 3. System framework

Figure 3 contains the main modules that formed the structure of the algorithm that will be implemented. How each of the modules work is summarized in the pseudo code presented next.

Hybrid recommender algorithm

```
Build the association rules

Calculate the similarity (Jaccard) between movie genre into
matrix C (movies x movies)

Create matrix A (users x movies) and populate with 0
Create matrix B (users x N)- this is the TOP N matrix and
at the end of the process will contain the recommendations

// compute the Top N matrix B
For each rule R
     For each user U
          For each movie ML from left side of rule R
               If user U has rated movie ML
                      Then for each movie MR from right side
                                                  of rule R
                         A[U][MR] += conf(R)

For each user U
     Select the top N values from A[U] into B[U]

For each user U
     For each movie M from B[U]
          For each similar movie MS from C[M] (Jaccard
                                            coefficient)
               If MS was not rated then add MS to B[U]

Return B
```

There are two initial operations that need to be performed. The first one is to build the association rules. The second step is to calculate the similarity between the movies based on their genre and store it in a similarity matrix (C).

The next step is to parse all the rules and for each user to build another matrix (A). This matrix will contain the confidence with which a movie can be recommended to a user. The confidence is obtained by adding the confidence of the rules where the movies recommended are part of the rule consequent (right hand sise of the rule). This confidence matrix is then used to select the top-N recommendations for each user based on the sums of the confidence. These will be stored in another matrix (B).

This matrix (B), along with the one obtained initially containing the movie similarity (C) will be used to expand the space of the recommendations. Movies that belong to exactly the same genre as the top-N movies stored in matrix B are added to the recommendation list. If enough movies are not generated in this way, movies that are closer based on the Jaccard coefficient of the genre vector can be added as well.

Chapter 4: Experimental design

4.1 Introduction

If the previous chapters described the recommender system from a theoretical point of view, this chapter will look at how theory is put into practice. The experimental configuration, the experimental design and the preprocessing that was performed on the data prior to the actual experimentation is described.

4.2 Experiment setup

The original layout of the ratings was not appropriate for building the association rules. A layout resembling the one of a market basket analysis was used. Each transaction corresponds to a user and each item to a movie. This is all represented in a matrix where the rows are the transactions (users) and the columns represent the movies. If a movie was rated positively by the user, then the value in this matrix corresponding to that user and that movie will be 1. Otherwise the value will be 0.

The same matrix layout was used in order to represent the movies that will be recommended. A similar matrix is built based on the values in the test subset and by comparing the two matrixes the performance metrics can be calculated.

Initially the rules were generated using the Weka environment (*www.weka.net.nz*). Weka is a machine learning workbench developed at the University of Waikato in New Zealand and is the most widely used open source machine learning toolkit worldwide.

Despite its wide usage, Weka proved to have some disadvantages for this exercise. The heap size could not be increased over the 1432 MB limit. This meant that the program crashed when it was trying to generate the association rules at lower support and confidence levels. Also the time needed to generate rules at low support and confidence level grew considerably.

The standalone version of ARminer, called ARtool (http://www.cs.umb.edu/~laur/ARtool), was used instead of Weka. The time needed to generate the rules decreased considerably when compared to the Weak implementation. Also it was possible to lower the support and confidence levels beyond what was possible in Weka. The layout of the output includes the support and confidence of each rule, along with the movies from the left and right sides of the rule.

The rest of the algorithm is implemented using C#. The hardware platform for all the experiments was a computer with a dual core 2 GHz processor and 1 GB of RAM memory.

The similarity between movies was calculated using the Jaccard coefficient between the genre vectors. The results are put into a matrix with the dimension (movie x movie).

The matrix containing user ratings is parsed more than once when the algorithm is run. The way data was represented was compacted in order to reduce the time needed to go through all the ratings. Because it was quite a sparse matrix, instead of representing both rated movies (with 1) and not rated movies (with 0) only the rated movies were represented. The first column contained the number of ratings and the next columns contained the IDs of the movies rated thus reducing the number of values the algorithm needs to read.

The same compression method was used for the Jaccard coefficient matrix. A matrix containing all the coefficients between all of the movies is calculated. Then, for example, a matrix for all movies that have a Jaccard coefficient greater than 0.75 is then created. This new matrix will not have as dimensions (movies x movies). Instead, the rows will represent each movie and the first column will contain the number of movies that have a coefficient greater than 0.75 with the movie on that row. The following columns contain the IDs of the corresponding movies. Thus the number of columns will be reduced to the maximum of the numbers contained in the first column when taken over all the rows.

The experiments that were run tested different values for the confidence and support parameters for their influence on precision and recall. The recommendations are represented in a binary matrix and so are the ratings in the test subset. This meant that the Jaccard similarity coefficient could also be calculated.

Also, different values for top N parameter were used and the impact on algorithm performance was assessed. The effect of increasing the algorithm coverage using the similarity between movie genres was the next step in the experimentation. The experiments presented in the next section were carried out. Their results are presented and analyzed in Chapter 5.

4.3 Experimental Design

The Weka environment was used in order to generate the association rules on the entire data provided in the u1 subset. The purpose of the experiment was to see if the implementation of a neighborhood was an option in the development of the hybrid system.

There were four experiments run in order to test the two hypotheses proposed in chapter 3. The first three experiments are related to the first hypothesis and test only the use of association rules in order to produce recommendations. The last experiment introduces the use of genre information and is intended to test the second hypothesis.

Experiment 1

In building association rules, two parameters can be altered: the support and the confidence of the rules. ARtool was used in these experiments in order to reduce the values for support and confidence as much as possible. How these values influenced the recall and the other metrics were recorded in order to choose the best combination for the next set of experiments.

Experiment 2

The second set of experiments conducted was similar to the first set. The difference is that the matrix with the user ratings for the movies was inverted. The same setup was used as in Experiment 1 set and the same parameters were varied.

Experiment 3

This set is based on Experiment 1. For the best level of support and confidence, only the top-N movies recommended were selected for each user. There were different settings of the N parameter that were tested: 10, 20, 30, 40 and 50. The purpose of this set of experiments was to find the effect of limiting the number of recommendations on the performance metrics. Also the results of these experiments are the baseline for assessing the performance of the innovative aspect of the hybrid recommender system proposed (i.e. the introduction of the genre similarity).

Experiment 4

The final set of experiments concentrated on the performance increase gained by adding the concept of similarity between movies. The same layout is used as in Experiment 3, with the variable being N (the number of movies recommended for each user). The same values for N are tested: 10, 20, 30, 40 and 50.

Chapter 5: Empirical study

5.1 Introduction

The previous chapter contained information about how the data was preprocessed. Also the purpose of the experiments and how they were set up to test our hypotheses were presented.

This chapter continues by presenting the actual results from each experiment. Based on these empirical results an analysis is performed comparing them to previous research and to what the algorithm was setup to do.

5.2 Experimental results

The first experiment that was run was generating association rules on the first training subset called u1. The results from this run proved that association rules need to be run on the entire dataset rather than just on a neighborhood. This conclusion was reached as the time needed to generate the association rules was very long. A characteristic of the dataset was identified after this experiment: the fact that it is quite sparse.

The experiments were run using the ARtool program for generating the association rules.

Experiment 1: The effect of support and confidence levels

We tested a wide range for the support and confidence parameters. The lowest values that could be used with the ARtool program for the given dataset were 0.04 for support and 0.75 for confidence. Table 5 contains the values for the Jacard coefficient obtained for all the five subsets.

support	confidence	u1	u2	u3	u4	u5	average
0.04	0.75	0.074568	0.060544	0.06091	0.062984	0.061801	0.064161
0.04	0.5	0.116918	0.102403	0.079381	0.07728	0.074163	0.090029
0.05	0.75	0.056878	0.04726	0.044842	0.052945	0.050097	0.050405
0.05	0.5	0.1085	0.096345	0.06091	0.0772	0.073163	0.083224
							0.071955

Table 5. Jaccard coefficient at different support and confidence levels

Table 6 contains the values for the precision at different support and confidence levels while Table 7 contains the figures obtained for recall.

support	confidence	u1	u2	u3	u4	u5	average
0.04	0.75	0.340419	0.312737	0.259529	0.2637	0.214202	0.278117
0.04	0.5	0.218403	0.178381	0.130903	0.123924	0.117881	0.153899
0.05	0.75	0.379752	0.361279	0.286918	0.323587	0.26687	0.323681
0.05	0.5	0.239668	0.195861	0.259529	0.145294	0.12764	0.193599
							0.237324

Table 6. Precision at different support and confidence levels

support	confidence	u1	u2	u3	u4	u5	average
0.04	0.75	0.089618	0.073874	0.07988	0.084023	0.092706	0.08402
0.04	0.5	0.245585	0.244374	0.227909	0.240905	0.249352	0.241625
0.05	0.75	0.063665	0.053001	0.053153	0.062221	0.063788	0.059166
0.05	0.5	0.193586	0.196006	0.07988	0.191295	0.201773	0.172508
							0.13933

Table 7. Recall at different support and confidence levels

An observation that can be made by looking at the figures obtained is that while precision is decreasing with the lowering of support and confidence, the Jaccard coefficient and recall are increasing. This happens because precision is closely related to the quality of the recommendations of the rules. If the rules have a relatively good confidence then the movies recommended from those rules are very likely to be liked by the users. On the other hand, if support and confidence are reduced, more recommendations will be generated by the rules but the chances for those movies to be liked by the users will decrease as a result. The F1 metric follows the trend of the Jaccard coefficient and recall as shown in Table 8.

support	confidence	u1	u2	u3	u4	u5	average
0.04	0.75	0.13161	0.107616	0.107087	0.110564	0.107821	0.11294
0.04	0.5	0.199774	0.175664	0.138385	0.135082	0.130112	0.155804
0.05	0.75	0.101835	0.084022	0.079976	0.092169	0.087308	0.089062
0.05	0.5	0.186868	0.167289	0.107087	0.135258	0.128447	0.14499
							0.125699

Table 8. F1 metric at different support and confidence levels

Experiment 2: Use of the inverted matrix

The next experiment that was run was in order to see if using the inverted ratings matrix will help recommend more movies to users. The values of the parameters are presented in Table 9 (Jaccard), Table 10 (precision) and Table 11 (recall).

support	confidence	u1	u2	u3	u4	u5	average
0.03	0.75	0.138494	0	0.126541	0.104043	0.093151	0.092446
0.03	0.5	0.114356	0.169222	0.110321	0.077452	0.060883	0.106447
0.05	0.5	0.061904	0.173484	0.115831	0.094814	0.072733	0.103753
							0.100882

Table 9. Jaccard coefficient at different support and confidence levels for the inverted matrix

support	confidence	u1	u2	u3	u4	u5	average
0.03	0.75	0.197163	0	0.209519	0.160547	0.120245	0.137495
0.03	0.5	0.158306	0.235367	0.134691	0.090227	0.068999	0.137518
0.05	0.5	0.063068	0.268949	0.132795	0.108691	0.080299	0.13076
							0.135258

Table 10 Precision at different support and confidence levels for the inverted matrix

support	confidence	u1	u2	u3	u4	u5	average
0.03	0.75	0.496599	0	0.372571	0.374494	0.462821	0.341297
0.03	0.5	0.560485	0.521923	0.661539	0.631886	0.660132	0.607193
0.05	0.5	0.853061	0.35281	0.605724	0.698448	0.732101	0.648429
							0.532306

Table 11. Recall at different support and confidence levels for the inverted matrix

If at first glance the results obtained are a good improvement, a closer look will reveal the reason for this improvement. When calculating the metrics only the users that have rated a movie in the test subset and the users that have been recommended a movie are taken into account. The number of such users is extremely low. Less than ten users were recommended movies in each of the five datasets. This led to the conclusion that this approach is not worth considering for this dataset.

Experiment 3: The Effect of the Top N parameter

The next set of experiments was run at the confidence (0.5) and support (0.04) levels that resulted in producing the best coverage. This was done by sacrificing confidence. The purpose of these experiments was to find the top-N recommendations for each user. The values for N were 10, 20, 30, 40 and 50. The top-N recommendations were chosen based on the added confidence of the rules in which the movies recommended were in the consequent (right) side of the association rules that applied for each user. A rule applied for a user if the user has rated all the movies in the antecedent (left) side.

The values of the performance metrics are presented in Table 12 (Jaccard), Table 13 (precision), Table 14 (recall) and Table 15 (F1 metric).

N	u1	u2	u3	u4	u5	average
10	0.07837	0.081303	0.068344	0.068655	0.070621	0.073459
20	0.097291	0.094802	0.07635	0.073518	0.070621	0.082516
30	0.109548	0.10092	0.078421	0.076251	0.073118	0.087651
40	0.114671	0.10219	0.07908	0.077262	0.074073	0.089455
50	0.117065	0.102403	0.079381	0.077281	0.074163	0.090058
						0.084628

Table 12. Jaccard coefficient at different values for Top N

N	u1	u2	u3	u4	u5	average
10	0.282318	0.230226	0.175475	0.171841	0.135928	0.199158
20	0.251272	0.201203	0.151338	0.141403	0.135928	0.176229
30	0.236708	0.186527	0.137769	0.13016	0.125774	0.163388
40	0.22423	0.178798	0.131485	0.124638	0.119814	0.155793
50	0.219352	0.178381	0.130903	0.123926	0.117938	0.1541
						0.169733

Table 13. Precision at different values for Top N

N	u1	u2	u3	u4	u5	average
10	0.113238	0.12938	0.121794	0.125945	0.174498	0.132971
20	0.16118	0.182881	0.170095	0.173043	0.174498	0.172339
30	0.204139	0.222734	0.202958	0.213725	0.21236	0.211183
40	0.231826	0.242411	0.223739	0.237961	0.240847	0.235357
50	0.244826	0.244374	0.227909	0.240905	0.249187	0.24144
						0.198658

Table 14. Recall at different values for Top N

N	u1	u2	u3	u4	u5	average
10	0.139018	0.141443	0.120249	0.120709	0.124429	0.12917
20	0.169985	0.163539	0.133846	0.129056	0.124429	0.144171
30	0.188976	0.173272	0.136992	0.133623	0.128547	0.152282
40	0.196534	0.175352	0.137927	0.135073	0.129985	0.154974
50	0.199986	0.175664	0.138385	0.135084	0.130121	0.155848
						0.147289

Table 15. F1 metric at different values for Top N

By increasing the value of N, the same trends that were observed when lowering the support and confidence for the association rules can be observed. While precision is decreasing, the other metrics are improving. Top-N movies are ranked based on the confidence of the rules in which they are in the consequence. Selection of only the top 10 movies ensured that movies that were recommended by many rules or by rules with high precision were used. This increases the chances that these movies will be liked by the users. On the other hand this approach limits the space of recommendation (the number of different movies recommended) which translates into a lower recall value when fewer top-N movies are selected.

Experiment 4: Effect of genre similarity

The final iteration in the process of developing a hybrid recommender system to expand coverage was done by recommending movies with a similar genre to the ones already recommended.

The experiments were run at the same support (0.04) and confidence (0.5) levels as the top-N experiments. The values of the parameters are presented in Table 16 (Jaccard), Table 17 (precision), Table 18 (recall) and Table 19 (F1 metric).

N	u1	u2	u3	u4	u5	average
10	0.02095	0.016449	0.012798	0.01074	0.010909	0.014369
20	0.019167	0.014869	0.012618	0.010618	0.010354	0.013525
30	0.019595	0.01508	0.012901	0.010956	0.010654	0.013837
40	0.019959	0.01532	0.013112	0.011266	0.010943	0.01412
50	0.020093	0.015357	0.013159	0.011305	0.011056	0.014194
						0.014009

Table 16. Jaccard coefficient at different values for Top N after finding similar movies

N	u1	u2	u3	u4	u5	average
10	0.026489	0.019686	0.016846	0.012779	0.012913	0.017743
20	0.022656	0.016967	0.016282	0.012363	0.011946	0.016043
30	0.022896	0.017102	0.016561	0.012678	0.012215	0.01629
40	0.023193	0.017325	0.016773	0.013007	0.01251	0.016562
50	0.023305	0.01736	0.016822	0.013047	0.012632	0.016633
						0.016654

Table 17. Precision at different values for Top N after finding similar movies

N	u1	u2	u3	u4	u5	average
10	0.277271	0.279827	0.307101	0.299712	0.312348	0.295251
20	0.412207	0.416454	0.420874	0.400272	0.428316	0.415625
30	0.491877	0.489574	0.47702	0.470735	0.491302	0.484102
40	0.529536	0.514072	0.508719	0.49533	0.521529	0.513838
50	0.542468	0.516981	0.513103	0.498061	0.528621	0.519847
						0.445732

Table 18. Recall at different values for Top N after finding similar movies

N	u1	u2	u3	u4	u5	average
10	0.040205	0.031662	0.024246	0.020767	0.021137	0.027604
20	0.036962	0.028701	0.023919	0.020555	0.020106	0.026049
30	0.037777	0.029102	0.024454	0.021206	0.020676	0.026643
40	0.038456	0.029556	0.024856	0.021798	0.021228	0.027179
50	0.038706	0.029626	0.024947	0.021872	0.021441	0.027319
						0.026959

Table 19. F1 metric at different values for Top N after finding similar movies

The influence of the N parameter is not as strong as it was in the previous experiment. This is due to the fact that many other similar movies are found that have the same starting point as the N movies found using association rules.

5.3 Analysis

The analysis is based on the four measures that were recorded: Jaccard coefficient, precision, recall and the F1 metric. A literature review found several papers that used the same MovieLens dataset. However, only two of them provided sufficient data in order to be able to perform a comparison. The first part of the analysis will compare the results obtained in each of the four experiments carried out in the process of developing the algorithm. Then a comparison with previous work will be presented.

The first decision that was made was to drop the selection of a neighborhood on which to apply the association rules. This decision was made after it was found through experimentation that the time to run the association rules generation application was prohibitive. If this process were to be repeated for each user an online real time application would be hard to build. Sarwar et al. (2000) have built a system that uses the neighborhood approach. The neighborhood is created using the cosine between the user preferences. This was also one of the solutions that were considered for this application along with an implementation based on Euclidian distance computed with the help of demographic user characteristics. On the neighborhood generated, Sarwar et al. (2000) apply two different methods, namely, association rules or the simple method of recommending the most frequent item. However they fail to discuss the response time of their system thus casting doubt on the real-time applicability of the system.

The next set of experiments that were run was aimed at finding the influence of support and confidence of the association rules on the measurement metrics. The Weka environment was dropped in favor of ARtool, a purpose-built application for association rules creation. The application builds the rules in two steps. The first step is to find the most frequent item sets based on a minimum support. This is the most time consuming task. Once the most frequent item sets are found, the association rules are generated based on a minimum confidence level.

The conclusion of this experiment was that in order to obtain a greater coverage the support needs to be lowered as much as possible. The lowest value that could be achieved was 0.04. Increased coverage can be translated into a better recall. Also coverage and recall can be improved by reducing the confidence of the rules. This however affects the precision of the recommendations generated.

Once it was established that lowering confidence and support helps to get better coverage the next focus was on increasing the coverage of the algorithm.

An idea presented by Lin (2000) was to try and find the association rules on the inverted ratings matrix. If finding association rules on the original matrix established relationships between movies, inverting the matrix will find relationships between users. An association rule for the original matrix (with users as rows) the rules are of the form:

$$Ma \vee Mb \vee ... \vee Mc \rightarrow Md \vee ... \vee Me$$

This can be interpreted as: if movies Ma, Mb, …, Mc are all liked by a number (expressed in percentage of the total number) of users higher than the minimum support set that was set, then with a probability equal to the confidence of the rule, movies Md, …, Me are also liked by that same set of users who liked the movies in the left side of the rule. Each side of the rule (antecedent and consequent) can have one or more movies.

For the inverted matrix the rules are of the form:

$$Ua \vee Ub \vee ... \vee Uc \rightarrow Ud \vee ... \vee Ue$$

This can be interpreted as: if users Ua, Ub, …, Uc all liked a number (in percentage of the total number) of movies higher than the minimum support set, then with a probability equal to the confidence of the rule, users Ud, …, Ue will also like the movies that the users from the left side of the rule liked.. Each side of the rule (antecedent and consequent) can contain only or more users.

The approach based on the inverted matrix was dropped because it was generating recommendation for very few users (less than ten for each of the subsets).

The next method of increasing the number of recommendations was to select the top N recommendations for each user and based on those to try and find similar movies. The top N recommendations produced better precision results for the lower values (10 or 20) but as it increased the values of the metrics became similar to the approach where all the recommendations were considered.

Based on the top N recommendations, movies that had exactly the same movie genre vectors were selected in a bid to increase coverage and recall. The precision dropped dramatically, by a factor of around ten. At the same time, recall doubled. Also, the F1 metric decreased by a factor of five. The results of finding similar movies proved promising as they increased the coverage of the algorithm. The issue was that there were many movies that had exactly the same genre vector. For example if there were at least 300 movies that shared the same genre with one of the top 10 movies recommended, then the count of recommended movies will increase with the new 300 similar movies that was found. Coupled with the fact that the dataset is sparse this can be an issue. The users might actually like some of the 300 new movies generated but if in the testing set they have only rated a couple more movies, then the precision of the algorithm will be severely affected.

The other approaches that worked on the MovieLens dataset had different ways of assessing the results obtained. Sarwar et al. (2000) use the F1 measure, however, Kim and Kim (2003) are only taking into account recall.

Sarwar et al. (2000) use a neighborhood approach and then apply association rules. Kim and Kim (2003) use multi level association rules. Both papers have noticed the sparsity level of the dataset and agree that the greater the number of rules used, the better the results obtained will be. Sarwar et al. (2000) have used 0.2 for the confidence minimum level and 0.02 for the support minimum level. Kim and Kim (2003) do not give details about the minimum levels, just that they're using the whole set of rules generated in order to counter the sparsity of the dataset.

Both papers use as a benchmark the top N association rules application that was applied in the process of developing this hybrid algorithm too. Sarwar et al. (2000) have similar F1 metric levels obtained to the results described in this chapter. Kim and Kim (2003) also compare their findings to the single level association rules algorithm. Their recall values are better than the ones obtained in the top N experiments in this chapter. This could be due to the usage of lower support and confidence levels than what was used in this study.

In terms of improving the results of the classical association rules approach, Sarwar et al. (2000) do quite well by finding values of up to 0.22 for the F1 metric. Because separate results for recall and precision are not presented, it is difficult to asses which of the two metrics increased or if they both increased. If we use the F1 metric as the sole measure of performance then the proposed hybrid algorithm does not improve on the classical approach.

In the case of Kim and Kim (2003), recall is the only metric presented, and this does not improve for the MovieLens dataset. If we use only recall as the only mean of comparison, then the hybrid algorithm developed compares much better that the one proposed in Kim and Kim (2003).

Chapter 6: Conclusion

6.1 Achievements

The first hypothesis to be proved in this research was that association rules alone cannot be used to produce a good recommender system. The results obtained in the experiments suggest that in order to obtain good results, the support level and the confidence level need to be very low. A low support level is difficult to achieve as it takes a lot of time to compute the frequent item sets. Lowering the confidence threshold of the rules generated means that the precision of the algorithm is also affected.

For these reasons there is a need to improve a system that only uses association rules. The second hypothesis referred to using the similarity between movie genre to obtain better results. As can be seen from the experiments conducted, if we use recall as the main measure, at it was done by Kim and Kim (2003), then this is improved. However this is done by sacrificing precision.

Having a better recall can be argued as being more desirable than having a better precision. As suggested by Herlocker et al. (2004), and also by McNee et al. (2006) there is more to a recommender system that having good accuracy. One of the suggestions made by these researchers is that coverage is very important in the case of recommender systems. A better recall can be translated as a better coverage. In this research we evaluated coverage by testing the recall value generated by our algorithm on a independent hold-out test dataset that contained known outcomes for movie preferences amongst users. As future work we propose that this evaluation is reinforced by soliciting feedback from users in a live system.

A major issue that was found with the MovieLens dataset was its very sparse nature. This meant that there was a need for more rules to be generated in order to get better coverage. The use of ARMiner did not allow us to go beyond the 0.04 support threshold. This could be a limitation that if surpassed could produce much better results. Finding similar movies to the ones recommended is also a way of increasing the coverage.

The use of a neighborhood was dropped because it was found through experimentation that it would take a lot of time to build association rules on each user's neighborhood. The use of neighborhoods would also help overcome the sparsity property of the dataset used.

6.2 Future work

Initially the idea was to use an ontology to find similar movies and expand the recommendation space. This was dropped in favor of using the similarity between the movie genre vectors. The initial ontology approach was dropped as an ontology was not found and it was difficult to create one.

Another idea to link movies is to use more information about them found online. Each movie contains a link to its entry in the imdb online movie database. Information that could be useful can be found in this database includes the synopsis of the movie, where it was filmed and the cast that it features. Also there is a ranking system of the movies that can be used to recommend highly rated movies.

From the cast information and the data available in the MovieLens dataset more information about why some movies are rated together can be found. This can mean that a user could likes certain movies if they have a particular actor in common in the cast. Recommending movies based on the actors can mean that the algorithm might perform well in terms of accuracy but it might not fulfill other requirements that recommender systems have according to McNee et al. (2006), such as diversity of the recommendations made.

References

Agrawal, R., Imielinski, T. & Swami, A. (1993). Mining association rules between sets of items in large databases. Proceedings of the 1993 ACM SIGMOD Conference. 22 (2). 207-216.

Ansari, A., Essegaier, S. & Kohli, R. (2000). Internet Recommendation Systems. *Journal of Marketing Research*. 37 (3). 363-375

Burke, R. (2002). Hybrid Recommender Systems: Survey and Experiments. *User Modeling and User-Adapted Interaction*. 12: 331-370

Changchien, S. V., & Lu, T. C. (2001). Mining association rules procedure to support on-line recommendation by customers and products fragmentation. *Expert Systems with Applications*. 20. 325-335

Cho, Y. H., Kim, J. K., & Kim, S. H. (2002). A personalized recommender system based on web usage mining and decision tree induction. *Expert Systems with Applications*. 23. 329–342

Demiriz, A. (2004). Enhancing Product Recommender Systems on Sparse Binary Data. *Data Mining and Knowledge Discovery*. 9. 147–170

Fu, X., Budzik, J., & Hammond, K. J. (2000). Mining Navigation History for Recommendation. *Proceedings of the 2000 Conference on Intelligent User Interfaces*. 106-112

Herlocker, J. L., Konstan, J. A., Terveen, L. G., & Riedl, J. T. (2004). Evaluating Collaborative Filtering Recommender Systems. *ACM Transactions on Information Systems*. 22(1). 5-53

Kim, C., & Kim, J. (2003). A Recommendation Algorithm Using Multi-Level Association Rules. *Proceedings of the IEEE/WIC International Conference on Web Intelligence*

Lam, S. K., & Riedl, J. (2004). Shilling Recommender Systems for Fun and Profit. *Proceedings of the 13th international conference on World Wide Web*. 393-402

Lawrence, R. D., Almasi, G. S., Kotlyar, V., Viveros, M. S., & Duri, S. S. (2001). Personalization of Supermarket Product Recommendations. Data Mining and Knowledge Discovery. 5. 11–32

Lin, W. (2000). Association Rule Mining for Collaborative Recommender Systems. Retrieved on 29 September 2008 from www.wpi.edu

Lin, W., Alvarez, S. A., & Ruiz, C. (2002). Efficient Adaptive-Support Association Rule Mining for Recommender Systems. *Data Mining and Knowledge Discovery*. 6. 83–105

Linton, F. Charron, A., & Joy, D. (1998). OWL: A Recommender System for Organization-Wide Learning. *AAAI Technical Report*. 8. 65-69

Massa, P. & Avesani, P. (2004). Trust-Aware Collaborative Filtering for Recommender Systems. *Proceedings of Federated International Conference On The Move to Meaningful Internet*. 492-508

McNee, S. M., Riedl, J., & Konstan, J. A. (2006). Being Accurate is Not Enough: How Accuracy Metrics have hurt Recommender Systems. *CHI 2006*. 1097-1101

Middleton, S. E., Shadbolt, N. R., & de Roure, D. C. (2004). Ontological User Profiling in Recommender Systems. *ACM Transactions on Information Systems*. 22 (1). 54-88

Miller, B. N., Albert, I., Lam, S. K., Konstan, J. A., & Riedl, J. (2003). MovieLens Unplugged: Experiences with an Occasionally Connected Recommender System. *Proceedings of the 2003 Conference on Intelligent User Interfaces.* 263-266

Miller, B. N., Konstan, J. A., & Riedl, J. (2004). PocketLens: Toward a Personal Recommender System. *ACM Transactions on Information Systems*. 22 (3). 437-476

Mobasher, B., Cooley, R., & Srivastava, J. (2000). Automatic Personalization Based on Web Usage Mining. *Communications of the ACM.* 43 (8). 142-151

O'Donovan, J., & Smyth, B. (2005). Trust in Recommender Systems. *Proceedings of the 2005 Conference on Intelligent User Interfaces.* 167-174

Tan, P-N, Steinbach, M & Kumar, V. (2006). Introduction to Data Mining, Pearson Education Inc, pp 74-75.

Pechenizkiy, M., Puuronen, S., & Tsymbal, A. (2005). On the Use of Information Systems Research Methods in Data Mining. Information Systems Development: Advances in Theory, Practice and Education. Springer. 487-499

Resnick, P. & Varian, H. R. (1997). Recommender Systems. *Communications of the ACM.* 40 (3). 56-58

Tran, T., & Cohen, R. (2000). Hybrid Recommender Systems for Electronic Commerce. *AAAI Technical Report*. 78-84

Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2000). Analysis of Recommendation Algorithms for E-Commerce. *Proceedings of the 2nd ACM conference on Electronic commerce*. 158-167

Schein, A. I., Popescul, A., Ungar, L. H., & PennockD. M. (2002). Methods and Metrics for Cold-Start Recommendations. *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*. 253-260

Spens, K. M., & Kovacs, G. (2006). A content analysis of research approaches in logistics research. *International Journal of Physical Distribution & Logistics Management*. 36 (5). 374-390

Vozalis, E., & Margaritis, K. G. (2003). Analysis of Recommender Systems' Algorithms. *Proceedings of the HERCMA*. 1-14