# Recommender Systems in use

*Postgraduate Student*

Doctor of Philosophy
University of Edinburgh
2009

# Declaration

I declare that this thesis was composed by myself and that the work contained therein is my own, except where explicitly stated otherwise in the text.

(*Postgraduate Student*)

*To my ...*

# Abstract

My abstract.

# Contents

# Chapter 1

## 1.1 Introduction

In our everyday life we are constantly exposed to recommendations in some form or the other. Examples of this can be via televised commercials, signs in the supermarket, or the personal recommendations from sales people of all trades. It is possible for the television and supermarket advertisers to make more personal recommendations by the use of demographics such as statistics of which age group that watches tv during certain hours. However, from the group of people that are exposed to these recommendations, it is possible that only a fraction will actually find the commercial relevant to their inherent needs or interests. What if there existed a way to target each individual user with personal recommendations? While we might have to wait a little with that in regard to the television and supermarket (RFIDs are moving us closer to a reality in this area), it does already exist on the Internet: recommender systems. Where recommendations from the recommender systems described in this paper distinguish themselves from the above mentioned methods, is that the recommender systems are personalizing their recommendations for each of their individual users. This is done by aquiring information about each users preferences and interests, both given directly and indirectly by the user itself. Directly by the user typing in relevant information to the system and indirectly by observing the individual users behaviour via the trail of technological footprints that is left behind when the user interact with the individual site. These footprints includes items bought, items viewed, what persons with similar interests have bought, what they search for, etc. All this information is then crunched in various ways depending on which type of recommender system is used, and recommendations can be presented to the user.

It is argued that we are moving away from an era of search based interaction and entering one of discovery. The meaning behind this idea is that we are moving away from knowing what we want and hence search for, and instead are shown new products or sites that we did not knew existed, or that we even needed. No matter what, recommender systems will continue to evolve and keep increasing their efficiency, possibly until they can tell more about you than you can yourself.

Reference this article: `http://money.cnn.com/magazines/fortune/fortune_archive/2006/11/27/8394347/index.htm`

Quote: "The effect of recommender systems will be one of the most important changes in the next decade," says University of Minnesota computer science professor John Riedl, who built one of the first recommendation engines in the mid-1990s. "The social web is going to be driven by these systems."

## 1.2 Collaborative Recommendations

The main idea of this approach for recommendations, is to base the recommendations of items on similar users or similar items.
This method of recommending is one of the most widely spread, as it is used both at Amazon, Netflix and similar high-value companies Kilde: Linden et al. 2003.
In this section, the different implementation techniques of CF recommender systems will be described and a list of pro's and cons, be giving towards the end. We will distinguish between

what is known as user-based and item-based recommendations Kilde: Recommender Systems, An Introduction. But before going is to these descriptions, a short section for the commons features will be explained

For both types of recommendations the basic problem can be formulated like this: For any given user and non-rated item pair, try to estimate the rating the user will give, for this item. The most common approach is therefore to define a user-item matrix as seen in the table below insert a table here

### 1.2.1 user-user based recommendations

The approach described here are recommending on the basis of what is known as peer-users (also called neighbors). This means users that have similar preferences in the past as the user the systems is trying to recommend items to. The idea is, for the item n, which a user called Alice not yet have rated, find peer-users that have rated item n and based on the this, compute the rating for item n for the Alice. This means that the job of the RS is to: 1) find peer-users with similar taste to Alice and 2) take the rating for the item from the peer-users and based on this, predict the rating for the item for Alice. To illustrate this, we can return to our rating matrix in matrix $R$. Here Alice have not rated item 4 and the task of the RS is then to predict the rating for item 4, based on the ratings of this item from peer-users, which is user2 and user4. Most commonly the is done, using the Pearson correlation coefficientKILDE?, which calculates the similarity between users.

#### Pearson correlation

Before describing the mathematical formulations, we need first to establish the basic terms required for the calculations. As described in section **??** a matrix of items and users are made. The users will in the following be denoted as $U = u_1, \ldots, u_n$ , the items will be denoted as $P = p_1, \ldots, p_n$ and $R$ for the $nxm$ matrix for ratings $r_i, r_j$ for $i \in 1 \ldots n, j \in 1 \ldots m$. The possible ratings a user can give is 1-5, where 5 being the the most-liked rating option. The similarity between user a and user b, giving the matrix $R$ is defined as follows:

$$sim(a,b) = \frac{\sum_{p \in P}(r_{a,p} - \bar{r_a})(r_{b,p} - \bar{r_b})}{\sqrt{\sum_{p \in P}(r_{a,p} - \bar{r_a})^2}\sqrt{\sum_{p \in P}(r_{b,p} - \bar{r_b})^2}}$$

This will produce the similarity of user a and user b to be XXX. This is a relatively high similarity and using the same formula for user a and user c, we also end up with a high similarity. It should be noted, that the formula takes into account that users tends to interpret a rating scale differently. Meaning that, some users tend to give a lot of high ratings, whereas other never rates anything with a 5. Now that we have found two users (b and c), that have rated items similar to user a, we are able to proceed and predict rating for the items, user a not yet have rated. According to kilde til RS, an Introduction, p16 one possible way of making the prediction is based on this formula:

$$pred(a,p) = \bar{r_a} + \frac{\sum_{b \in N} sim(a,b) * (r_{b,p} - \bar{r_b})}{\sum_{b \in N} sim(a,b)}$$

Based on these predictions, we are now able to fill out the rating matrix for user a and making a list of top recommendations, based on this matrix.

The above presentation of a recommender systems is of course a very small example and in real world applications, the case of recommending is not a straight-forward as the method described here. The example just holds 4 users and 5 items, whereas real-world applications often contains tens of thousand or even millions of both items and users, which was the case for the Netflix problemkilde netflix problem. As we will later present, a problem about data

sparsity with rating for items, is also a very present and problematic issue for recommendations systems. Problems with scalability

### 1.2.2 item based recommendations

Another technique of collaborative recommendations, called item-based, are in certain areas very similiar to user-based technique described above, but has some advantages in the sense of the large dataset an real-world application consists of today. The main idea of this approach is to calculate the similarity between items, instead of users, based on ratings for the items. If we want to compute the rating for item5 and we return to our matrixMATRIX, we can see that the ratings given for item5, is close to the ratings given for both item2 and item4. Because of this similarity, we can say that: Since user a gave a rating of 4 to item2 and user b gave a rating of 3 to item4, item2 and item4 is similar in theirs ratings, a item-based recommender system will compute the average of these similar items ratings and give item5 a rating between 3 and 4. Bedre eksempel. p19 In the following a description of one item-based algorithm will be presented.

#### Cosine similarity measure

In the previous sections, we looked at how item5 had similar ratings as item2 and item4.Because of a small dataset, it was possible to observe this similarity without any kind of calculations. It is rarely the case that this is possible in real-world applications and one of the methods for calculating this similarity is to use the cosine similarity measure. Making a vector from each of the items, based on their ratings, this method uses the angle between the two vector for measuring of their similarity. The formula for calculating this angle is found below:

$$sim(\vec{a}, \vec{b}) = \frac{\vec{a} \cdot \vec{b}}{|\vec{a}| * |\vec{b}|}$$

explanation of the formula

The cosine similarity values goes from -1, trough 0 and to 1, where 1 indicates a strong similarity between the two compared items, just like the Pearson method, as indicated on the picture below:

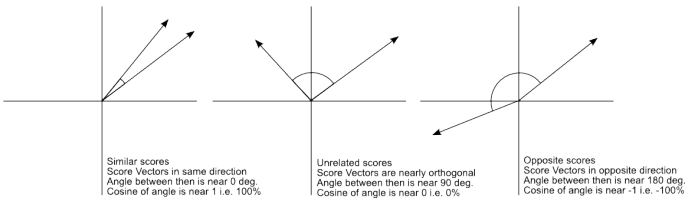


Figure 1.1: source: http://pyevolve.sourceforge.net/wordpress/?p=2497

Differently from the Pearson method, the formula does not take into account that users use a rating scale differently. maybe note the adjusted cosine measure her? p19 After the similarity between the items have been calculated, the recommender system can predict a rating for user a's item5, based on the ratings for the items similar to item5. This prediction is done, using the formula below:

$$pred(u, p) = \frac{\sum_{i \in ratedItems(u)} sim(i, p) * r_{u,i}}{\sum_{i \in ratedItems(a)} sim(i, p)}$$

#### Data sparsity and cold start problem

section: Data preprocessing

### 1.2.3 Advantages and disadvantages

According to kilde: Recommender systems, an introduction collaborative recommender systems is one of the most researched techniques of the two and because of this many different types of collaborative recommendations exists, as described in the above. Even with many different types, it is possible to outline some common advantages and disadvantages, which we be done in this section. Firstly, collaborative filtering can work independent of the items the system holds, since it uses ratings to recommender from, as opposed to content-based, which requires the item to have some content that can be compared kilde The next generation of recommender system, p18(nederst). more advantages? Likewise with content-based recommender method, the collaborative method do also have some disadvantages, or more correct, some challenges, described here as: New user problem, new item problem and data sparsity.

- New users are both problem for both methods, since both relies on some previous history about the users. In order for a recommender system to make recommendations for a user, the system must know something about the user preferences, which is historic ratings in a collaborative system.
- New items are frequently being added to a systems 'inventory'. This creates a challenge for a collaborative based recommender system, since it relies upon ratings for the items it recommends, and which it obviously does not have for new items.

## 1.3 Content-based Recommender Systems

### 1.3.1 The functionality of a content-based RS

The overall reason to implement a Content-based recommender system is basically the same as for other recommender systems; to deliver a list of recommendations that statistically will be seen as valuable to the user. Where it differentiates itself from the others is in the way these recommendations are generated. A content-based recommender system will try to recommend items to the user similar to items that the user has either bought or somehow interacted with in the past. So, in other words, the way the content-based recommender system approaches the recommendation process is by analyzing a set of descriptions of the previously mentioned items which has already been rated or interacted with by the user, and from this information build up a model around the users interests. The item preferences from this model will then be compared with those items from the Represented Items repository (see picture). The result returned from the system is a mathematical indication of the relevance these objects or items will represent for the user. As an example, this could help filter search result for webpages. If a resulting webpage has a negative score in comparison to the users interests, it will simply be removed from (or not added to) the list of recommendations.

### 1.3.2 The architecture and modules of the system

In this section we will describe the different components which substantiates the content-based recommender system by creating the user profile, comparing the user profile with the items contained in the system, and finally by choosing and presenting the recommended items to the user. There are three main steps used in the recommendation process, they are described below as follows; content analysis, profile learning, and filtering (see picture).

- Content Analyzer: When data has no apparent structure to it, such as text, a processing step which can extract the relevance from the material in a structured way is needed. The main responsibility of the Content Analyzer is to represent data (items, webpages, articles, documents, etc.) in such a way, that it can be used by the next processing steps. This is done by shifting the content representation from its original form to a form usable by the target module (this could for instance be a webpage represented as keyword vectors) via a specific extraction technique.

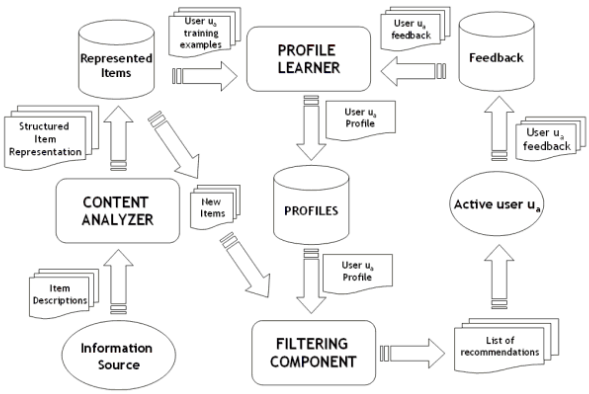


Figure 1.2: source: http://pyevolve.sourceforge.net/wordpress/?p=2497

- Profile Learner: This module collect and utilize the relevant information that has been analyzed and altered into a usable format by the content analyzer. In this case, relevant information is data which represents the users preferences. The profile learner aspires to put this information into the right order, and then use this ordered data to create the user profile for the individual user.
- Filtering Component: This module uses the profile representation in question and matches this onto the items in the Represented Items Repository. These items will then be further sorted in relation to the users interests and preferences. The result is generated either as a binary yes or no, or as part of a continuous examination of the item, meaning that perhaps there is not enough information on the user available in order to give a correct judgement of the item, so it is instead put on a "waiting list" until further information on the user can be analyzed.

These three components work together to recommend items to the user via a series of steps. These steps are illustrated at the picture below. The first step of the process occurs when the Content Analyzer receives a series of item descriptions from an information source. These descriptions are analyzed in order to extract specific features, such as keywords, from the unstructured text, and end up with a structured representation of the items. These items are stored in the repository Represented Items. In order to create and update the Profile for an active user (Ua) for which the recommendations are required, the system collects information regarding the users behavior in relation to certain items. This information is then stored in the repository Feedback. During the Profile Learner step, the above mentioned information in Feedback is processed together with the represented items in order to predict the relevance of these in relation to the user. A user can likewise create a profile where his or her area of interest are directly provided, thus making the Feedback part of optional relevance. There are basically two types of relevance feedback, positive and negative. The positive feedback indicates features that the user are interested in, whereas the negative feedback is an indicator of features that have no interest to the user. These types of feedback can be recorded via two different feedback techniques, namely implicit and explicit feedback. Explicit feedback refer to when the user actively makes a decision, such as a rating or evaluation, about a specific item. Implicit feedback refers to when the user does not make any active involvement, and is captured by monitoring and analyzing the user's activities and behavior. The explicit feedback technique is the easiest way for the Recommender System to make recommendations, since it gives a clear and actual indication of the users preferences. When speaking about the explicit kind of feedback, there are three main approaches: like/dislike, rating, and text comments. The first option works as a binary scale where the user can either choose to like or dislike an item. The rating option gives the user a larger scale to rate an item on, this could be from 1-10. The last option presents the user with a set of text comments from other users in order to aide the user in the decision-making whether to buy or not buy a specific item.

All of the above mentioned actions will be stored in the Feedback repository in order to help the Profile Learner create a more comprehensive picture of the users preferences. An advantage of the explicit feedback is that it simplifies the interpretation process of the Recommender System for the specific items in question. A disadvantage can be that the user is required to make an active choice, hence incurring a cognitive load on the user. The implicit feedback is centered around actions that do not require direct user involvement on a cognitive level, but more on their actions in relation to the items in question, such as saving, discarding, bookmarking, etc.

In order for the Recommender System to create the User Profile for user Ua, the system must first create a training set from the current repository of reference items. The training set is then used by the Profile Learner to match up against user Ua's preferences in order to build a solid foundation for the user profile. Future items in the Items repository will be matched against the user item preferences via the Filtering Component and, if there is a match, these items will be shown as recommendations to the user.

This training set of recommendations will need to be updated on a regular basis in order to keep the recommendations as close to the users current preferences as possible. This is of particular importance since a users preferences are most likely to be in a dynamic state of change over time.

### 1.3.3 Keyword-based Vector Space Model

The mathematical way in which most content-based recommender systems defines the relevance of an item is by the Vector Space Model, which is a spatial representation of text documents. In this model, each of the documents is represented by a vector in a n-dimensional space where n represents the number of terms or keywords from that particular document that matches a, for the system, overall vocabulary. In simple terms, each vector is giving a weight to each keyword to show how big a relevance or association these specific keywords has to the document in question. When using VSM to represent the weight of keywords in documents, there are two things that is important: weighting the keyword and measure the similarity or importance of the document in relation to the keyword. The weighting scheme most commonly used is TD-IDF weighting, which stands for

### 1.3.4 Advantages and disadvantages

When looking at recommender systems from a content-based systems point of view there are several advantages over a recommender system in a collaborative-based form.

The three main areas of advantage are user independence, transparency, and new items. The user indepence shows itself in that the content-based recommendations are solely based on the active users own preferences and does not rely on the interaction of others. The content-based recommendations are transparent in the way that all its recommendations are clearly based on the list of items placed on the users list of preferences, and consists of no estimated guesses. The advantage of a content-based recommender system in regards to new items that enters the Represented Items repository, is that these items will not need to be rated prior to being recommended to the users. Here, the only dependency is whether the item is similar enough to the individual users preferences.

As well as the above mentioned advantages, a content-based Recommender System also has some disadvantages. The three main areas here are limited content analysis, over-specialization, and new users. The limited content analysis revolves around the fact, that content-based Recommender Systems most often are dependant on domain knowledge. An example of this could be for a movie recommendation. Here, the Recommender System would need to know the actors, director, genre, and alike. If the Content Analyzer does not find a suitable amount of information, then it will be virtually impossible for the Recommender System to distinguish between items correlating with the user preferences and items that are of no interest at all. The disadvantage with over-specialization is that the user will never be recommended any

items in a positive unexpected manner. What is meant by this is that the users personal area of preferences might be somewhat wider than the area of the current information held in the User Profile, but since the recommended items for the individual user is based on a match with the items previously rated by the user, there will never be a recommendation deviating from this. This disadvantage is also referred to as The Serendipity Problem. The new user disadvantage can be sort of a predicament for the Recommender System. The issue here is that in order to give the new user a fair recommendation based on user interests and preferences, the user will need to have rated a certain amount of items for the Filtering Component to compare with. When the new user first create his or her account, the recommender system is somewhat forced to either recommend a set of random items or recommend what the average user would be recommended. In most recommender systems the latter is the solution of choice.

Where is it used

Use maybe Amazon, and give an example on the occurence of a recommendation, for instance for a book, based on another book that the user has read

How -The mathematical background for the recommendations (different approaches)

Classical and advanced techniques

Different types of algorithms and their function

Trends and directions (what role does user generated content play!?!?)

Next generation of content-based RS

Conclusion

Buzz words:

Serendipitous recommendations Surprisingly interesting recommendations that the user might not have thought of or discovered otherwise.

Extraction techniques

Information Retrieval systems

-

## 1.4 Hybrid approaches

So far descriptions of collaborative and content-based recommender systems have been given in section insert sections refs here. These two have, of course, both advantages and disadvantages as we also have described earlier and both do also have challenges. To overcome some of these challenges, one could combine the approaches in to a 'hybrid'. This could for instance be in a case, if a system have a rich community of users and the items in that system could be compared by keywords, then recommendations could be made by combining the features of content-based recommendations with peer-users, which, in theory, could result in higher quality recommendations. In the following a list of applied hybrizing-approaches will be presented, to outline the different methods for doing this: According to 'Towards the next generation of recommender p20'kilderef! the methods can be classified as the following four: "(1) implementing collaborative and content-based methods separately and combining their predictions, (2) incorporating soe content-based characteristics into a collaborative approach, (3) incorporating some collaborative characteristics into a content-based approach, and (4) constructing a general unifying model that incorporates both content-based and collaborative characteristics". It should be noted here, that this list is probably not comprehensive and many, many other hybridized approaches do exists. The first methods is to implement the collaborative and content-based independently of each other and the obtain a recommendation for the same items from both and the afterwards either combining the rating from each system into a single rating or selecting the 'best' rating relative to the recommended item based on a predetermined quality. The second method is to add some content-based characteristics to a collaborative recommender system. Here the system uses the 'Profile learner' to compare the similarity between users and thereby partly overcomes some of the sparsity problems, since it is common that users-pair often does not have many rated in common. Another feature of this approach is that users can be recommended an item based not only if its highly rated by peer-users, but also just if this item matches the users profile. Third method somewhat does the opposite of the second

one. This method uses collaborative characteristics to NOT FINISHED. Need explanation of the last two approaches

# Chapter 2

# First Appendix