NTNU

Norwegian University of Science and Technology

Design and Evaluation of a Recommender System for Course Selection

Hans Fredrik Unelsrød

Master of Science in Computer Science

Submission date:	June 2011
Supervisor:	Anders Kofod-Petersen, IDI

Norwegian University of Science and Technology
Department of Computer and Information Science

Abstract

In this thesis we will construct a recommender system for course selection in higher education (more specifically, at The Norwegian University of Science and Technology). Some of what makes our approach novel compared with existing solutions is that we weight each user in the collaborative filtering process based on their chosen degree subject (major) and wether or not the two users being compared are friends. Also, we utilize both collaborative filtering and content-based recommendations in a hybrid solution. Another novel aspect of our solution is that we construct our system on top of an existing website designed for rating courses. This gives us unique access to a dataset containing thousands of user-ratings of courses.

Acknowledgments

I would like to thank the Classmate team for allowing me to use all of their data and the use of their users as test users. I would also like to thank the administration at NTNU for helping me get a sample from their data on the structure of degree programs. The Classmate users also deserve to be recognized for their help in evaluating the system. And of course Anders Kofod-Petersen for his help in guiding me through this project.

Hans Fredrik Unelsrød
Trondheim, June 5, 2011

Contents

List of Figures

List of Tables

Chapter 1

Introduction and Overview

Using recommender systems to filter vast amounts of information has been done successfully in many production systems [MacManus, 2009]. The most popular domains to use recommender systems in are movies, books and other entertainment media. We wanted to use those same techniques used to recommend movies to recommend university courses. There are quite a few similarities between the two domains; there is a wast number of different options to choose from, and all of the options have both well defined quantitative characteristics as well as less defined qualitative characteristics. There are however aspects of course selection that make it quite different from choosing a movie to watch on a Sunday night, we will discuss these differences in more detail later.

1.1 Motivation

Students get to choose between hundreds of courses every time they want to take a course. In this case having many choices is a good thing, but it does make it hard for a student to wade through and read all of the information on each course. Universities usually employ guidance counselors, people who are tasked with helping students making their choice. But in practice the counselors are often overloaded with too many students and not enough time, and some students are not satisfied with the level of knowledge that the counselors have [Young, 2011].

The author is involved in developing a website called Classmate that gives students the ability to rate and review all of the courses they have taken. Through his involvement with Classmate the author has access to all of the data associated with the website, this was another motivating factor for doing this project. We will discuss Classmate in more detail in Section 3.1.

1.2 Goals and Research Questions

Goal 1 Develop and evaluate a recommender system for course selection that utilizes a hybrid of both collaborative- and content-filtering.

The end goal of this thesis is to have developed a recommender system that furthers the state-of-the-art in recommender systems for course selection. By that we mean that we will have developed a recommender system that attempts to use existing recommender system algorithms in a way that has not been done with recommender systems for course selection. Our plan is to make a recommender system that uses a hybrid of both collaborative filtering and content based recommendations, as well as attempting to use the social graph between the users and other factors to determine the correlation between users.

Existing course recommender systems have not been widely documented in research papers, but from what we were able to find there has not been a course recommender that uses a hybrid of both collaborative filtering and content filtering.

Goal 2 Develop a recommender system that uses the users‘ social graph and their chosen degree subject as signals in the collaborative filtering algorithm.

We believe that including additional signals into the collaborative filtering process can enhance the accuracy of the results. Course selection is not solely based on personal taste or preference, it is heavily influenced by other factors like prerequisites, number of credits, etc. We will discuss these factors in more detail in Section 3.8. Our plan is to weight each users correlation with each other by using these signals.

Goal 3 Develop a recommender system that is good enough to be used in a production environment (i.e. Classmate.no).

The system has to be good enough (fast, scalable, accurate) for the users of Classmate.no to want to use it. This means that it has to be able to recommend courses in real-time, and the majority of the users of Classmate have to be satisfied with the recommendations they get.

Research question 1 How will building upon a website for course-rating impact the recommender system?

In order to come up with good course recommendations you have to have both a thorough knowledge of the structure of the degree programs, as well as knowledge of all the courses that the user has taken and how he/she feels about those courses. This means that in a ’naive’ system the user has to enter a lot of information before the system can give good recommendations.

Most of the explicit data input used by this system will come from what users are already entering into Classmate without the promise of getting anything in return. By not forcing the users to fill out long and potentially tedious questionnaires we hope to make the system more pleasant to use and creating a sort of network effect where the more users we have the more complete the data will get and the better the recommendations get. Will building on something like Classmate be enough to alleviate this problem?

Much of the previous research done on recommender systems for course selection has had to make due with small data-sets. We are able to build on an existing platform that already has over a thousand users that have entered over 4.000 ratings. How will this impact the development of the recommender system?

Research question 2 How well will including additional signals into the collaborative filtering process work?

In Goal 2 we discussed how we want to integrate the users‘ social graph and their chosen degree subject as signals into the collaborative filtering process. As far as we know this has not been tested before, how much will it affect the performance of the collaborative filtering process?

Research question 3 How will students respond to using a recommender system for course selection?

Choosing a university course could potentially have huge consequences, it could be the difference between getting or not getting your dream job, it could be the difference between being allowed into other courses that you want to take in the future or not, and it is a big commitment time-wise that you can't regret after the course has started (or at least not after a certain amount of time). These factors make recommending a course different from recommending entertainment items to purchase (like books). Will the users trust an automated recommender system to give them good recommendations?

1.3 Thesis Structure

The rest of this thesis is structured as follows. In Chapter 2 we will describe how we went about doing a survey of existing literature on recommender systems for course selection and recommender systems in general. In Chapter 3 we will go through the background of topics that are relevant to this thesis. In Chapter 4 we will describe and explain how the recommender system we have developed works. Then in Chapter 5 we will evaluate the system. Finally in Chapter 6 we will discuss the results and how they relate to the goals and research questions we defined in the previous section.

Chapter 2

Literature Survey

2.1 Method

In this chapter we present a structured survey of the literature exploring recommender systems for course selection, explanations in recommender systems as well as recommender systems in general. This chapter is organized as follows; first we define the goals for doing the literature survey, then we define a search protocol, finally we look at how we chose the studies we did and how we evaluated the quality of the studies.

2.1.1 Goals

The first step of our literature survey was to define the goals for doing a literature survey. These goals are separate, but may be overlapping, from the over all goals and research questions that we formalized in Chapter 1.2.

Goal 1 Give us an overview of the different types of recommender systems.

Goal 2 Find out how the different types of evaluation methods compares.

Goal 3 Give us an overview of how different types of explanations work in relation to recommender systems.

Goal 4 Give us an overview of existing work in recommender systems for course selection.

2.1.2 Sources

The next step was to gather a list of sources that were likely to provide relevant studies. A more thorough literature review might do a complete survey just to find sources, we did not find this to be necessary. The sources we decided on using where chosen because they all provide studies covering a wide range of topics and from a wide range of sub-sources.

2.1.3 Search Terms

The next step was to define search terms. All of our sources provide search engines that are capable of sorting results based on relevance, the method used to define relevance differs from source to source. For Google Search relevance is based on PageRank, i.e. how many other sites link to this site. For Citeseer X, relevance is based on how good of a match the paper is and how

Source	Type	URL
Google Search	General Search Engine	http://google.com
Google Scholar	Academic Search Engine	http://scholar.google.com
IEEE Xplore	Digital Library	http://ieeexplore.ieee.org/
Citeseer X	Digital Library	http://citeseerx.ist.psu.edu/

Table 2.1: Literature sources

many citations it has. IEEE Xplore and Google Scholar does not make it clear how they come up with their rankings, but presumably they both utilize similar metrics to Google Search and Citeseer X.

Search Terms:

- Recommender System
- Recommender System Evaluation
- Course Recommender System
- Explanation Recommender System
- Collaborative Filtering Recommender System

2.2 Results

Some of the search terms we used are too broad to return a manageable set of results, for instance searching for "recommender system" on Google Scholar returns 23.400 results, so we rely somewhat on the ranking done by the individual search engine. We kept evaluating papers, by reading their abstract, until we found enough good papers to meet our goals for doing this survey. After selecting the papers to read we read through them and discarded the ones that we either considered to be low quality or that did not help us achieve our goals for doing the survey.

For each result we looked at we evaluated the title and abstract, if the abstract matched our inclusion criteria we moved on to reading the entire text. Our inclusion criteria where:

- The study‘s main concern is recommender systems
- The study‘s topic is helpful in completing our literature survey goals

In addition to the papers found through our structured search we also supplemented with papers recommend by others as well as some papers referenced in other papers. There are papers out there on this and related topics that we wish we could have covered, but we feel that the ones we chose where enough to fill the goals we set out to complete when we started this survey.

Search Term	Source	Number of. results
Recommender System	Google Search	247,000
Recommender System	Google Scholar	23,400
Recommender System	Citeseer X	103,000
Recommender System	IEEE	1,300
Recommender System Evaluation	Google Scholar	17,200
Recommender System Evaluation	IEEE	160
Recommender System Evaluation	Citeseer X	431,000
Recommender System Evaluation	Google Search	136,000
Course Recommender System	Google Scholar	9,900
Course Recommender System	IEEE	26
Course Recommender System	Citeseer X	175,000
Course Recommender System	Google Search	8.300,000
Recommender System Explanation	Google Scholar	9,500
Recommender System Explanation	IEEE	13
Recommender System Explanation	Citesser	170,000
Recommender System Explanation	Google Search	301,000

Table 2.2: Literature search results

Author	Title	Citation
Weber, Roger and Schek, Hans-Jörg and Blott, Stephen	A Quantitative Analysis and Performance Study for Similarity-Search Methods in High-Dimensional Spaces	[Weber et al., 1998]
Michael P. O'Mahony and Barry Smyth	A recommender system for on-line course enrolment: an initial study	[O'Mahony and Smyth, 2007]
Tintarev, N. and Masthoff, J.	A Survey of Explanations in Recommender Systems	[Tintarev and Masthoff, 2007]
KoKang Chu, Maiga Chang, Yen-Teh Hsia	Desining a Course Recommendation System on Web based on the Students' Course Selection Records	[KoKang Chu, 2009]
John S. Breese and David Heckerman and Carl Kadie	Empirical Analysis of Predictive Algorithms for Collaborative Filtering	[Breese et al., 1998]
Jonathan L. Herlocker and Joseph A. Konstan and Loren G. Terveen and John and T. Riedl	Evaluating collaborative filtering recommender systems	[Herlocker et al., 2004]
Mustafa Bilgic	Explaining Recommendations: Satisfaction vs. Promotion	[Bilgic, 2005]
S¿rmo, Frode and Cassens, Jörg and Aamodt, Agnar	Explanation in Case-Based Reasoning—Perspectives and Goals	[Sørmo et al., 2005]
Paul Resnick and Neophytos Iacovou and Mitesh Suchak and Peter Bergstrom and John Riedl	GroupLens: An Open Architecture for Collaborative Filtering of Netnews	[Resnick et al., 1994]
Sarwar, Badrul and Karypis, George and Konstan, Joseph and Reidl, John	Item-based collaborative filtering recommendation algorithms	[Sarwar et al., 2001]
Rosta Farzan and Peter Brusilovsky	P.: Social navigation support in a course recommendation system	[Farzan and Brusilovsky, 2006]
Symeonidis, P. and Nanopoulos, A. and Manolopoulos, Y.	Providing Justifications in Recommender Systems	[Symeonidis et al., 2008]
Cesar Vivalardi, Javier Bravo, Leile Shafti, Alvaro Ortigosa	Recommendation in Higher Education Using Data Mining Techniques	[Cesar Vivalardi, 2009]
Upendra Shardanand and Pattie Maes	Social Information Filtering: Algorithms for Automating "Word of Mouth"	[Shardanand and Maes, 1995]

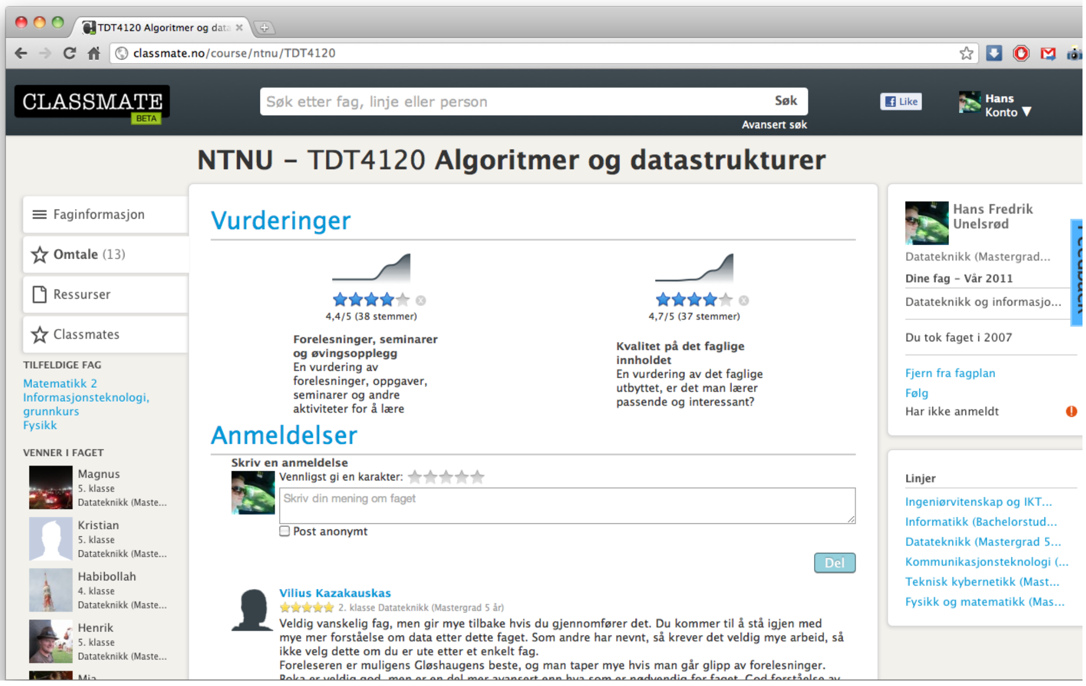
Table 2.3: Papers included in survey

Chapter 3

Theory and Background

3.1 Classmate

Classmate.no is a website where students can rate and post reviews of all of the courses that NTNU teaches. Classmate is organized into course-pages, one page for each course, see Figure 3.1. Each course-page contains a set of ratings, reviews and all of the static information that the school provides about that course. Each course can be rated on either the total impression of the course, quality of the lectures or the quality of the curriculum. Each review consists of a rating as well as a textual description of the user‘s impression of the course.



NTNU – TDT4120 Algoritmer og datastrukturer

4,4/5 (38 stemmer)

Forelesninger, seminarer og øvingsopplegg
En vurdering av forelesninger, oppgaver, seminarer og andre aktiviteter for å lære

4,7/5 (37 stemmer)

Kvalitet på det faglige innholdet
En vurdering av det faglige utbyttet, er det man lærer passende og interessant?

Vilius Kazakauskas
2. klasse Datateknikk (Mastergrad 5 år)
Veldig vanskelig fag, men gir mye tilbake hvis du gjennomfører det. Du kommer til å stå igjen med mye mer forståelse om data etter dette faget. Som andre har nevnt, så krever det veldig mye arbeid, så ikke velg dette om du er ute etter et enkelt fag.
Foreleseren er muligens Gløshaugens beste, og man taper mye hvis man går glipp av forelesninger.
Boka er veldig god, men er en del mer avansert enn hva som er nødvendig for faget. God forståelse av

Figure 3.1: Screenshot of a course page on Classmate

When Classmate was launched (in the middle of January 2011), the site was only open to registered users. Users who did not have an account could only see the landing page. In order for a user to get an account they had to have both a Facebook account and an '@stud.ntnu.no' email address. Having such strict requirements on user accounts was done to make it harder for spammers/trolls to get access. Requiring that all users connect their Facebook account also means that we have a social graph between the users, without them having to enter all of their friend relationships all over again. In May of 2011 Classmate was relaunched, the new version of Classmate included a new design, a notification system and pages for each degree program. The new version also changed how the registration process works, after the relaunch, Classmate was opened up to unregistered users and the registration requirements where changed. Users could now register with either a Facebook account or an '@ntnu.no' email address.

After registering, each user is given a profile page where they can enter their degree subject and which class year they are currently in. The users can also add courses to their course history. This course history is not only visible to the user himself, but also to other users of the system. When a user adds a course they are asked to enter what year and which semester they took the course as well as which parallel of lectures they followed, and why they took the course. The reason for taking the course gives us quite a lot of information; the user can choose between obligatory, elective, perspective and other. By combining all of the enrollment data we are able to deduct which courses are obligatory for which major and at which class year. This information is crucial for the filtering part of our course recommender.

Getting the structure of all of the degree programs from somewhere else, like the NTNU.no webpage, would be very hard since each faculty has its own website with no relation to the other faculties. We also tried contacting the NTNU administration, they did give us access to the structure of one degree program. The file NTNU gave us contains the structure of he Master in Computer Science program for students graduating in 2011. A degree program at NTNU is improbably complex, the file describing the structure of the computer science program is 50,260 lines long.

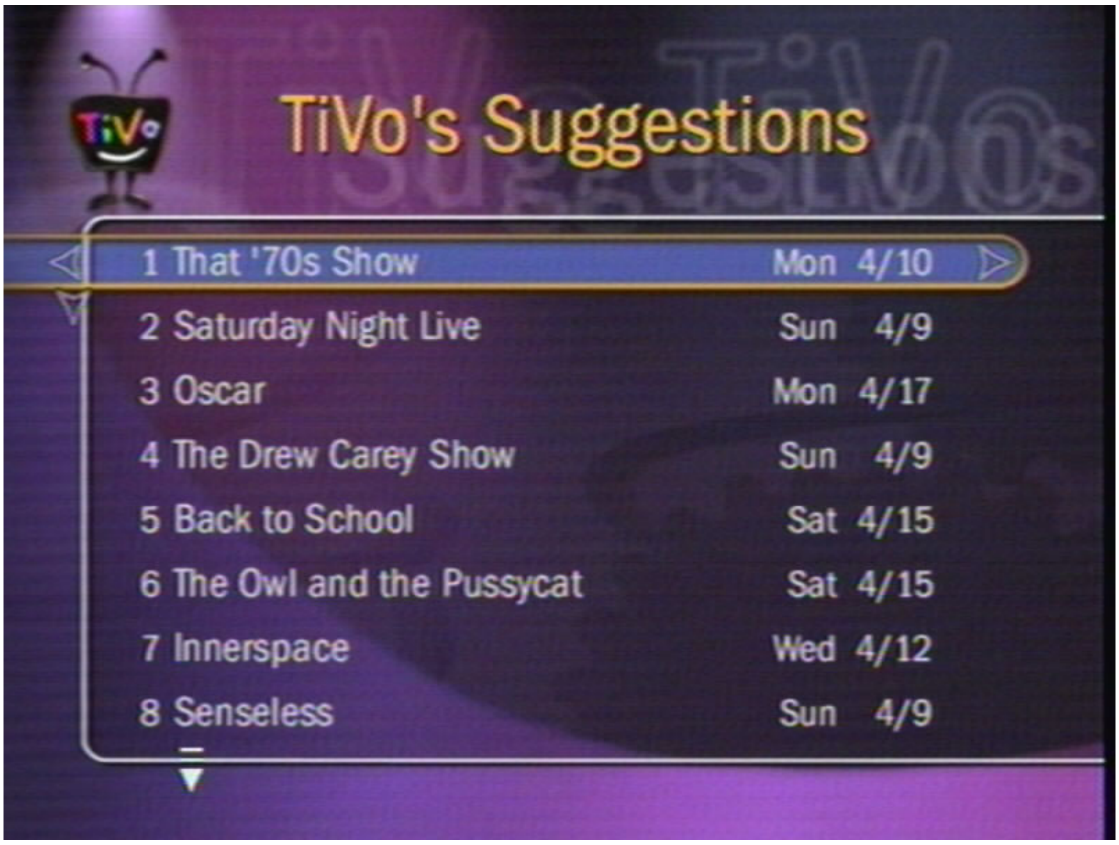
The Classmate team therefore decided to rely only on crowd-sourcing, since it is more likely to give a broad and up-to-date view of the degree structures for the most relevant degree programs. Classmate uses the data they get from each user‘s course history to estimate which courses are obligatory, elective and taken for other reasons by students at a given degree program. The users can also manually edit the courses related to a degree program, so if the algorithm gets it wrong the hope is that a user will spot the mistake and correct it.

By having the users enter their data through Classmate it hopefully is a more pleasant process than if we were to give the users a big questionnaire that they had to fill out when they wanted a recommendation. Users seem to be willing to enter a lot of data into Classmate, if for no other reason than to let other people see which courses they took or just because it feels good to have a complete dataset. Even before we launched any of the recommender system features, users had added an average of 3 courses each. And after the redesign all new users start out with a profile that includes all of the courses we know are obligatory for their degree subject, thereby increasing the number of connections between users and courses. By the end of May the average number of courses in each user profile was roughly 10.

3.2 Recommender Systems

A recommender system is an information filtering system technique that attempts to recommend information items (books, movies etc.) that are likely to fit a user‘s taste. A recommender system can be used to filter any sort of information, and with any sort of preference. Arguably the most well known examples of recommender systems in use today are Amazon‘s ”recommended items” and the TV-show recommendation system in the Tivo DVR (see Figure 3.2). The main motivation for developing recommender systems is the ever increasing information overload in today‘s society. We can all relate to being overwhelmed by the seemingly endless supply of books, articles, movies, TV-shows, games, etc. that we want to consume. Recommender systems can, as mentioned, be used to filter any sort of information, be it tourist destinations or research papers. Or as we will discuss in this thesis, recommending university courses to students.

The first recommender system was Tapestry [Goldberg et al., 1992], its goal was to filter through discussion postings on an online bulletin system to decide which ones where worth reading.


TiVo's Suggestions

1 That '70s Show	Mon	4/10
2 Saturday Night Live	Sun	4/9
3 Oscar	Mon	4/17
4 The Drew Carey Show	Sun	4/9
5 Back to School	Sat	4/15
6 The Owl and the Pussycat	Sat	4/15
7 Innerspace	Wed	4/12
8 Senseless	Sun	4/9

Figure 3.2: Screenshot of the ”Tivo Suggestions” interface

The goal of a recommender system is to be able to look at a large collection of items and select the items that are most likely to give the user the most satisfaction. Thus a recommender system has to be able to figure out what features a user likes and also be able to categorize each element in the collection by the same features. There are two main approaches to solving this problem, Collaborative Filtering (CF) and Content-Based Recommender Systems (CB) [Adomavicius and Tuzhilin, 2005]. In general CF matches a user with items by looking at the history of actions the user has taken (this could include both explicit and implicit actions). One of the most popular ways of implementing CF is to look at users that have a similar rating history to the user that the system is giving recommendations to, and then suggest items those users have rated highly. CB systems on the other hand builds a model of the user, a user profile, and then matches that user profile with models of the items in the dataset. A user profile can be built using both explicit and implicit data. Examples of explicit data include rating of items, ranking of items and asking the user questions. Examples of implicit data include social relationships, analyzing the items the user views or other behaviors. We will explore the details of both CF and CB based recommender systems later.

There are several user-tasks that a good recommender system needs to perform well in order to fully satisfy all users. The main goal is of course to find the items that the user will be the most satisfied with, but it is also important to give that information context in the form of an explanation that tells the user why the selected items where selected. The importance of finding all the items that the user could like (low rate of false negatives) varies between different scenarios. In a recommender system for books on Amazon.com it is unfeasible to show all the books available that the user might like. But in a recommender system that finds precedents for legal cases for instance, it is extremely important for the user to not overlook any possible cases. Other uses where a recommender system could aim at providing a good experience for is users that enjoy browsing through suggestions for no other reason than to browse, users that like to try to manipulate the system, users that like to help others and users that like to influence others[Herlocker et al., 2004].

3.3 Collaborative Filtering

Collaborative Filtering (CF) is one of the most widely used techniques in recommender systems [Resnick et al., 1994; Shardanand and Maes, 1995]. CF works by constructing a database of connections between items and users. When a user requests a recommendation, the system looks up items that other users with similar preferences have expressed a preference for. What constitutes a similar user will depend on the system and the amount of data available. Typical parameters used to find similar users are rating history, purchase history or more implicit factors like links they have clicked on.

In order to make accurate predictions we need to have a good way of calculating the correlation between users. There are two algorithms which are typically used for this [Resnick et al., 1994; Salton, 1989]; Pearson correlation and Cosine-based correlation.

The Pearson correlation coefficient between two variables is defined as the covariance of the two variables divided by the product of their standard deviations [Rodgers and Nicewander, 1988].

$$\rho_{X,Y} = \frac{cov(X,Y)}{\sigma_X \sigma_Y} = \frac{E[(X-\mu_X)(Y-\mu_Y)]}{\sigma_X \sigma_Y} \tag{3.1}$$

The Pearson correlation measures the strength of linear dependency between two lines. It is independent to changes in location and scale. Cosine-based correlation uses the cosine similarity between two vectors. Cosine similarity is calculated by measuring the cosine of the angle between the vectors. If the angle is 0 the cosine similarity is 1, it is less than 1 for any other angle.

Cosine-based correlation works by the same principles as Pearson correlation [Salton, 1989]. We will discuss how cosine-based correlation works in more detail in Section 3.4.

Several algorithms for solving the problem of Collaborative Filtering have been proposed. The main categories of CF algorithms are the memory-based algorithms and the model-based algorithms. memory-based algorithms use the entire user-item dataset to find a set of similar users, known as neighbors, it then finds the items that these neighbors preferred. The most widely used type of memory-based CF algorithms are nearest-neighbor. The nearest-neighbor algorithm simply finds the k users that are the shortest distance from the user that the system is giving recommendations to. The metric used to calculate the distance between each user can vary from implementation to implementation, the most basic being euclidian distance.

The next step is to predict the rating that the user we are giving recommendations to would have given the courses that the closely correlated user have rated . We do this by weighting the sum of the ratings of the other users.

$$p_{a,j} = \overline{v}_a + k \sum_{i=1}^{n} w(a,i)(v_{i,j} - \overline{v}_i) \tag{3.2}$$

> Where n is the number of users in the collaborative filtering database with nonzero weights. The weights w(i, a) can reflect distance, correlation or similarity between each user i and the active user. k is a normalizing factor such that the absolute values of the weights sum to unity[Breese et al., 1998].

The other category of CF algorithms is what is called model-based CF. The difference is that with model-based CF the system creates a model of all users in an off-line stage, it can then use that data to easily look up similar users in the online stage.

3.4 Content Based

Content based filtering is the other main category of recommender system approaches. It attempts to find items to recommend by analyzing the characteristics of the items. The goal is to find items that have similar characteristics as the items that the user has shown a preference for. This makes it able to recommend items that no user has shown a preference for (e.g. a new item), something that collaborative filtering is not able to do.

Content filtering comes from information retrieval [Balabanovic and Shoham, 1997]. An important step in creating a good content filtering system is to tag items with relevant feature values. This can be hard if the items have few quantitative features. In Table 3.4 we see an example of all of the features we have available for a course in our dataset.

> A common way of defining both the user and item profiles are as vectors of weights [Adomavicius and Tuzhilin, 2005]. For a user u we have wu = (wu1, . . . , wun), and for an item i we have wi = (wi1,...,win) where n is the number of features or keywords used in the system. Dependent on domain of the recommender system, the weights in wu reflect users‘ preferences of features or keywords, and the weights in wi reflect items‘ level of inclusion of the same features or keywords.
>
> Measuring the utility of an item for a user in content-based filtering is often done using heuristic functions. A commonly used heuristic for this purpose is the cosine similarity metric which is also used in collaborative filtering for measuring user-user similarities [Salton, 1989]:
>
> $$utility(u,i) = cos(wu, wi) = \frac{w_u \cdot w_i}{\|w_u\|_2 * \|w_i\|_2} \tag{3.3}$$
>
> where u is a user, i is an item, $\vec{wu}$ is the content weight-vector of user u and $\vec{wi}$ is the content weight-vector of item i. Other approaches build models based on underlying training data using machine learning techniques such as Bayesian classifiers, decision

Feature	Value
Course Code	TDT4100
Name	Object-Oriented Programming
Language	Norwegian
Year	2010
Units	7.5
Level	Basic
Lecture hours	4
Practice hours	7
Extra hours	1
Responsible unit	Department of Computer and Information Science
Description	Basic algorithms and data structures, constructs and control flow in object-oriented languages. Modularization and re-use. Standard application programmers interface (API). Unit testing, error detection and tools for this. Object-oriented design. Use of class, sequence and collaboration diagrams in the UML. Use of design patterns. Java is used as implementation language.
Learning goal	The students will have skills in programming, training in usage of relevant programming methods and tools, Also knowledge and understanding of usage areas, restrictions and underlying theory.
Learning methods and activities	Lectures, exercise lectures, individual exercises and project work in groups. The project aims at creating a computer game. If there is a re-sit examination, the examination form may change from written to oral.
Course coordinator	Hallvard Trætteberg
Lecturer(s)	Hallvard Trætteberg, Trond Aalberg
Subject area	Technological subjects

Table 3.1: Featureset for a course

trees and artificial neural networks [Adomavicius and Tuzhilin, 2005]. [Lillegraven and Wolden, 2010]

3.5 Hybrid

Some recommender systems combine the results from collaborative filtering and content-based filtering. There are several ways to combine these results:

Weighted	A final utility value of an item for a user is calculated by combining the weighted outputs from several independent recommender techniques.
Switching	Dependent on conditions, one of several recommendation techniques is selected to generate the recommendation.
Mixed	Recommendation techniques independently generate recommendations which are presented together.
Feature combination	Use collaborative information as additional features of items in a content-based approach.
Cascade	Let one recommendation technique filter out a candidate set of items which is refined by a second technique before recommending to user.
Feature augmentation	Use one technique to create additional features or ratings which can be used by second technique to create recommendations.
Meta-level	Let one technique build a model which is used as an input to the second technique which generates recommendations.

[Lillegraven and Wolden, 2010; Burke, 2002]

3.6 Evaluation Methods

3.6.1 Recommender Systems In General

Before setting out to evaluate a recommender system it is important to have a look at what the recommender system is trying to achieve[Herlocker et al., 2004]. There are several different end-user goals that a recommender system could be focused on achieving. As we briefly mentioned in the description of recommender systems some systems focus on finding all viable suggestions, while others focus on only finding a few, but good, results. Other factors to look at are the sequence of recommendations, if the results will just be used for browsing (and not to make big decisions) and if the system is meant to influence others. Understanding the context in which the recommender system will be used is important when deciding how to evaluate a recommender system.

Another important choice that has to be made when evaluating a recommender system is whether to use an existing dataset or to create a synthetic one. Research has shown that testing a recommender system with synthetic datasets tend to not give realistic results, synthetic datasets often end up giving the algorithm being tested an advantage compared to other algorithms [Herlocker et al., 2004]. This is because researchers tend to bias the dataset at the type data that fits the algorithm being tested. Synthetic data should therefore only be used in limited cases, like an early step of development to verify that the algorithm is performing as expected, or when doing relative comparison of the performance of a recommender system when tuning parameters.

When looking for a dataset to use for evaluation it is important to consider all of the features that describes a dataset. [Herlocker et al., 2004] found that the main features of a recommender system dataset are:

- The content topic being recommender/rated and the associated context in which rating/recommendation takes place
- The user tasks supported by the recommender
- The novelty need and the quality need
- The cost/benefit ration of false/true positives/negatives
- The granularity of true user preferences

As mentioned, one of the most common domains for recommender systems are entertainment content (movies, music, etc.). Some of the features that characterize this domain is that the consequences of making a wrong choice are low, and a big portion of the users prefer a high degree of serendipity. With serendipity we mean a recommendation that was unexpected, almost random, but still very much in line with what the user likes.

This contrasts with the domain we are looking at. In course selection the costs associated with making a wrong choice are huge, while the preference for serendipity varies a lot from person to person. Only recommending courses that a user has knowledge of will not do them much good, they are more likely to be looking for a new courses that they have not heard about, otherwise they would not have much use for the recommender system. But we should not dismiss the usage of the recommender system in giving validation to a choice of a known course.

3.6.2 Collaborative Filtering

In order to evaluate the CF part of the recommender system, it is necessary to define an accuracy metric. An accuracy metric empirically measures how close a recommender system‘s predicted ranking of items matches that of the user [Herlocker et al., 2004]. An accuracy metric can be used to compare one system with another, or to compare different versions of the same system. For our purposes the most important use for an accuracy metric is to compare different versions of our own system. The goal of this thesis is not to develop a CF correlation algorithm that is better than what is already out there, our goal is to use an existing CF correlation algorithm to give better course recommendations.

So what we need is an objective way of measuring the quality of recommendations with the different options we have when selecting a distance function for the k-nearest algorithm (correlation). We will also use this metric when setting the weighting for friend relationships between users, the same goes for weighting of users who chosen the same degree subject. Accuracy metrics for recommender systems can be divided into three categories; Predictive, Classification and Rank [Herlocker et al., 2004]. Predictive measures how accurate the predicted rankings of the recommender system matches those of the user, this is only useful for systems that strive to correctly predict the rating that the user would give each recommender item. Classification metrics measure the frequency with which a recommender system makes correct or incorrect decisions about whether or not to recommend an item. Rank accuracy metrics measure the recommender systems ability to produce a correct ordering of recommended items.

One of the biggest challenges when selecting an accuracy metric for collaborative filtering is that there is no single accepted standard. So researchers have continued to introduce new metrics when they evaluate their systems. For our purposes we will not focus on comparing the performance of our recommender with other course recommender systems. There are too many factors that impact the performance of a course recommender system, the dataset being the main one. We were not able to find any good reference datasets that we could utilize. But we do have to measure the performance of our recommender system in different configurations. For this we can use an accuracy metric, the most widely used accuracy metric for this purpose is precision/recall [Herlocker et al., 2004].

Precision and recall

Precision and recall measures two sides of classification accuracy [van Rijsbergen, 1979]. Precision denotes the number of correct classifications (true positives) in relation to the total number of classifications (true positives and false positives). Recall denotes the number of correct classifications (true positives) in relation to the total number of correct cases (true positives and false negatives). Precision and recall are often inversely correlated, for instance a search engine that wants to increase the number of true positives will also be including more false positives, thus increasing recall, but reducing precision.

3.6.3 Content Based

Content-based filtering can also be evaluated using precision and recall. But doing so does not make much sense. In order to measure precision and recall we need to know if the user actually would prefer the courses we recommend. In order to measure the quality of the CF predictions we recommended courses even if the user had rated them in the past, by splitting the data into a training and a test set. This made it possible to see whether or not a recommended course was highly rated by the user. Doing the same thing for CB recommendations would not yield as good of a result. The courses recommended by the CB step are recommended due to their similarities with courses we know the user has preferred in the past, so there is no reason for the user to have rated any of the courses recommended by the CB step. So we can not split the data into training and test sets. Therefore CB is best evaluated by doing user evaluations, asking a group of users how well the suggestions match with their actual preferences.

3.7 Explanations

A recommender system is only as good as the users think it its. Even if a recommender system could achieve perfect accuracy it would not mean much if the users could not be convinced that the recommendations where right. User satisfaction and derivatives of user satisfaction such as serendipity, diversity and trust are increasingly seen as important [Tintarev and Masthoff, 2007]. The effectiveness of an explanation can be measured in two fundamentally different approaches; promotion and satisfaction. Promotion is how well the system is at convincing the user to go for the recommended items. Satisfaction is how well the system is at informing the user as to why a particular item was recommended.

For a lot of recommender systems promotion might be the main goal, for instance when Amazon recommends books to its users. But it can be argued that satisfaction is more important in the long run. For course selection we can certainly say that satisfaction is the most important goal. We have absolutely nothing to gain from a user taking a course they end up not liking.

Different types of explanations have different goals. Showing how the recommender system works, transparency, could help convince the users that the recommendations are given based on sound reasoning and increase the users‘ trust in the system [Tintarev and Masthoff, 2007; Sørmo et al., 2005]. Explaining why the answer is a good answer could help provide justification, and increase confidence in the advice of the recommender system. Teaching the user about the domain could help increase understanding of recommendations in a domain where the user is not knowledgable.

From the point of view of a user the goal of the explanation is not necessarily to understand how the system solved the problem (i.e. how it came up with the recommendations that it did), but rather why a given recommendation was recommended.

3.8 Course selection

There are a lot of different deciding-factors that play in to the selection process for courses during college/higher-education. [Marsh, 1984] found that the teaching-style and the teachers ability to give good lectures are one of the most important factors that make a student like or dislike a course. But in reality there are several other factors that are likely to play into which course a student selects. From the authors own perspective as a student, as well as anecdotal evidence from fellow students some of the most important of these are (in no particular order):

Prerequisite	The course might be a prerequisite to a course or degree specialization that the student wants to enroll in at a later point.
Interest	The course might be about a topic which the student wants to learn more about, be it for personal interest or professional need.
Career goal	The course might be a good fit with the type of job they want to get after they are finished with their study.
Social situation	Many students would prefer to take a course where they have friends also taking the course, that they can work with on assignments, etc..
Difficulty	Sometimes students choose a course just because it is an easy way to get credits.
Type of exam	Some students do not like courses where 100 % of the grade is determined by how well they do at a written exam, they might be more inclined to choose a course where the work done in the class counts towards the final grade.
Timetable	Some courses do not fit into a students timetable, making them impossible to choose. Other times a student might prefer to find a course that is late in the afternoon so they can sleep in (not the most noble of reasons, but it is a reality for some).
Course format	The number of units/credits, number of lectures per week and other organizational features related to a course is another factor that might be important in course selection.

Getting an overview over all available courses can be challenging, at NTNU for instance there are over 3,000 courses available [NTNU, 2011]. The only information the school puts out is a short title and description together with some facts about when and how the course is taught. In order to get more information about a course a student can either enroll in the course and get access to its website, talk to a student that has previously taken the course or contact a counselor who might have some knowledge of the course. This makes navigating the jungle of courses difficult. In addition, there are no functionality to filter the courses on the NTNU website, this makes it very hard to find all of the courses that are potential choices.

3.9 Recommender Systems in Course Selection

The main use-case for recommender systems has, as mentioned, been in filtering of entertainment options. As far as we where able to find, usage in course selection has been minimal; [Farzan and Brusilovsky, 2006; Cesar Vivalardi, 2009; O'Mahony and Smyth, 2007].

In [Farzan and Brusilovsky, 2006] the authors describe how they developed CourseAgent, a recommender system for course selection focused on finding courses that matches the career goals of the user. One of the most novel parts of the CourseAgent system is the "do-it-for-yourself" approach to getting users to provide feedback to the system.

In [Cesar Vivalardi, 2009] the authors construct a course recommender system using data mining techniques. This paper focused on classifying courses into "likely to succeed" or "not likely to succeed", denoting wether or not the student is likely to get a good grade in the course based on his grade history in similar courses. The authors had access to quite a lot of data, their university provided them with a database containing over 100.000 records of student enrollments, including the grade the student got in that course as well as the students GPA up to the point of taking the course. Evaluation of the system showed that it was able to achieve roughly 80 % accuracy using the final year of the dataset as test data and the remaining data to teach the system. We believe there are several weaknesses with this system, first and foremost is the fact that they only take into account wether or not the student is likely to get a good grade, they do not look at wether or not the student is likely to enjoy the course or wether or not the course will be useful for the student. Secondly the system only classifies courses into yes and no, the system does not rank the courses.

Another interesting case of using a recommender system for course selection was published in an article in The Chronicle just as we were finishing this thesis [Young, 2011]. Apparently undergraduates at Austin Peay State University were invited to try out a new recommender system before having their scheduled meetings with their academic advisers. According to the article:

> When suggesting a course, the automated system considers each student's planned major, past academic performance, and data on how similar students fared in that class. It crunches this information to arrive at a recommendation. An early test of the system found that it could lead to higher grades and fewer dropouts, officials say.

The article goes on to state that the students at the university were not satisfied with their advisers, the advisers could not devote enough time to each student and did not have sufficient knowledge of all the courses offered at the school. Ian Ayres, a Yale Law School professor interviewed in the article, states that "Humans tend to have blind spots when handling tasks like advising, which involve complex systems. People often give too much weight to certain details based on personal preferences."

Mr. Denley, the man responsible for developing the system, states that "I don't think the major thrust will be to push people to classes that are sort of easy A's. I hope the major effect will be instead to open students' eyes to courses that they were dimly aware of.". In line with our goals for this system. Mr. Denley finished off with a choice quote: "If eHarmony works well, why not this?".

Chapter 4

Solution

Our solution was built on top of the existing code of classmate.no. Building on top of a website that gives something back to the user in return for their input and then using that input to give recommendations has been attempted before [Farzan and Brusilovsky, 2006]. The approach of CourseAgent is similar to our approach in that it attempts to make the process of providing feedback a separate activity from the recommendation process. In CourseAgent the users are given a career planning interface that shows them how much progress they have made towards their career goals. We get the impression that CourseAgent is still a recommender system at heart. We believe that our approach to build on top of Classmate, which is a course ratings/review system at heart, will make the users even more willing to give feedback to the system. We are of the opinion that the CourseAgent paper lacks a thorough user evaluation, as far as we can tell they only tested the system on 15 students. Given the existing system we are building on, we where fortunate enough to have a fairly large dataset from the outset. This meant we could run tests on that dataset to fine-tune our Collaborative Filtering algorithms.

Classmate already had a data model with a data set when we started work on this thesis, so we could start doing data mining of the existing data right away. One of the first things we wanted to find out was how much data we had to work with. The number of users at the time of starting this work was 519. Out of these 519 users 150 of them had entered what degree program they are in and 158 of them had entered which class year they are in. The number of students for each class year was roughly evenly distributed between the first and the fifth (last for most degree programs) class years. The 519 users had added a combined total of 2434 courses to their course histories, roughly 5 per user. And they had entered in 1775 ratings, roughly 3.4 per user. The database contained all of the courses at NTNU at the time, 3144 courses. So the number of users and ratings was actually less than the number of courses. So if the ratings were evenly distributed we would have had 0.56 ratings per course to work with, in practice the ratings were distributed over 297 courses with 1 or more ratings.

The fact that the data was so sparse meant that we had to focus on doing more than just collaborative filtering. As we will se later, we utilized a hybrid approach where we use both collaborative filtering and content based filtering to get the best of both worlds. The Collaborative Filtering part gives us recommendations based on what other seemingly similar students and friends of the user prefers. Content Based filtering on the other hand, gives us recommendations that are similar to courses the user has shown a preference for in the past.

4.1 Collaborative Filtering

We started out by developing the Collaborative Filtering (CF) part of the recommender system. CF gives us serendipity. It may recommend courses that the user has no apparent relation to the courses he typically chooses (even though there is a relation). And it enables us to recommend courses based on courses preferred by people in the user‘s social graph (which is an important part of our recommender system). As discussed earlier, there are two main types of CF algorithms. Memory based CF uses rating data to compute similarity between users or items, while model based CF uses machine learning to find patterns based on training data, these patterns are then used on real data when making suggestions to a user.

The advantages of memory based CF is that it is easy to extend the weighting to include for instance additional signals, memory based CF has been proven in practice, and it is easier to construct good explanations based on memory based CF. The weaknesses of memory-based CF is that it does not work as well on sparse data sets. [Sarwar et al., 2001] showed that model based CF can improve scalability, but at the cost of accuracy. We chose to use memory based CF, mainly due to its ability to easily explain recommendations and its track record in production systems.

4.1.1 Initial Step

The first step in memory-based CF is to find the users that have the most similar rating history to the user we are giving recommendations to. This requires that we calculate the correlation with every single other user in the system. Doing this in real-time is not feasible, especially not as the number of users (and ratings for each user) continue to grow. Therefore we decided to do this step off-line. This requires us to do O(N log N) correlation calculations. So it takes quite a while, around 15 minutes for 500 users, but this is easy to run off-line for now. In the future this might turn out to be a scalability problem, but trying to make the system scalable beyond the foreseeable future is out of the scope of this paper. Each correlation value is stored in a database for retrieval during the on-line step. The off-line correlation step goes through all of the users, for each user we need to calculate the correlation with every other user (except the ones we already have calculated the correlation to, this is what make the complexity O(N log N) instead of $O(N^2)$).

The complexity of this could obviously lead to problems with scaling, we will discuss this in further detail later. In practice our algorithm uses approximately 2 hours to run through all of the 1.100 registered with Classmate at the time of writing.

In order to find the correlation between two users there are, as mentioned, two algorithms that are typically used, Pearson Correlation and Cosine-based correlation. In addition to these two we also experimented with using the absolute difference between each rating weighted with the number of ratings.

In order to determine which of these algorithms best suited our needs we needed to measure the accuracy of them. In order to measure accuracy we need to know the correct classification of all of the test cases. This is obviously not possible with the way our recommender system usually works, it is set up to only recommend courses that the user has not rated. Thus we don’t know wether or not a recommended courses is a course the user would have liked. In order to get around this we separated the data-set into training and test data.

We used two different training sets, one consisting of 50 % of the data, and on consisting of 75 % of the data. Since we plan to have a limit to the number of results we return (and we expect this limit to be hit in the absolute majority of cases), we needed to turn off the limit during testing. In the cases where we could not find any results we set the precision and recall both to 1, arguably we could just as well have set it to 0, the important thing is to be consistent in all of the trials. As mentioned earlier, precision and recall should only be used as a relative measure of performance, not an absolute measure of the recommender system.

Correlation Algorithm	Training Data Set Proportion	Precision	Recall
Custom Distance Function	50 %	0.908	0.860
Pearson Correlation	50 %	0.911	0.861
Cosine Distance	50 %	0.906	0.865
Custom Distance Function	75 %	0.873	0.863
Pearson Correlation	75 %	0.905	0.864
Cosine Distance	75 %	0.855	0.948

Table 4.1: Precision and recall for different correlation algorithms

For our purposes precision is most important, we do not plan on presenting the user with an exhaustive list of options, the goal of our system is to give good and unexpected suggestions. Therefore we will focus on comparing precision for now, but we will discuss a way to compare precision and recall using a single metric later. Using our distance metric as the baseline we can see that Pearson Correlation resulted in .63 % higher precision, while Cosine distance resulted in .27 % lower precision than our custom distance metric.

It has been shown that precision and recall are inversely corelated [Cleverdon et al., 1966] and are dependent on the number of results returned to the user. There are several approaches to solving this issue, one of them is the F1 metric. The F1 metric combines precision and recall into a single number [Makhoul et al., 1999].

$$F1 = \frac{2PR}{P + R} \tag{4.1}$$

Correlation Algorithm	F1
Cosine Distance	0.885
Pearson Correlation	0.885
Custom Distance Function	0.883

Table 4.2: F1 metric for different correlation algorithms

The F1 metric shows the same trend as the precision metric did. Pearson has the best precision and recall, Cosine distance is second and our custom distance function is last. Since the differences between the different correlation measures were so low we did not do any more exhaustive evaluation of them and decided on using the Pearson Correlation.

In addition to calculating the simple correlation, we also weight the correlation based on wether or not the users we are comparing are friends with each other and wether or not they are in the same degree program. Initially users who are in the same degree program were weighted 20 % higher, while friends were weighted 10 % higher. These numbers where initially just guesses, and they both started out at 20 %. But after the initial survey we found that users showed a lower preference for selecting courses based on just the fact that a friends was in the course.

In order to find more accurately how heavy the friend relationships and common degree subjects should be weighted, we used the same precision and correlation measurements we used to choose how to measure the distance between users. We started out by getting a baseline measure without giving friend relationships or common degree subject any weight. Then we proceeded by running tests with different values. We ran all of theses tests on the full dataset, therefore it took roughly one hour to run the tests on our system. In hindsight we realize that it might have been beneficial to limit the size of the dataset during these tests. It would have allowed us to run much more thorough tests. Although it could be argued that the variance in precision and recall is so low that finding the perfect weights for our current dataset would not make much difference.

Friend Weight	Degree Subject Weight	Precision	Recall
1	1	0.911	0.861
1	1.25	0.911	0.863
1	1.5	0.918	0.866
1.25	1	0.915	0.863
1.5	1	0.872	0.864
1.25	1.25	0.872	0.866
1.5	1.5	0.874	0.864

Table 4.3: Precision and recall for different signal weights

These tests told us that putting too much weight in the fact that two users are friends does not work well with our data. It also told us that focusing on users with common degree subjects does improve the accuracy. We decided to not use the F1 metric here, since we only wanted to focus on precision. For the record, recall stays roughly the same in all tests.

We can also see that when both signals are highly weighted the accuracy goes down, this could be due to the fact that we apply the degree subject factor first, so the relative impact of the friend factor ends up being too high. In the final system we ended up going for a highly weighted degree subject (1.5). We had to abandon the idea of weighting the correlation based on friend relationships after both the user evaluation and the metric tests showed worse results than expected. The reason we did not test with any higher weighting of degree subject is that we wanted to keep the serendipity factor, which naturally declines the more we increase the weighting for common degree subject.

4.1.2 Online Step

The on-line step in the collaborative filtering part of the recommender system starts by finding the k-nearest neighbors of the user we are giving recommendations to, user A. It uses the data found in the off-line step to find the nearest users. Then we find the courses that user A is most likely to rate highly, as described in more detail in Section 3.3.

Lastly the courses recommended by the CF part of the recommender system is added to the list of courses to be excluded from recommendation in the CB part.

4.2 Content Based

CB recommendations, as previously discussed, are based on similarities between items (i.e. courses) and the preferences of the user (either explicit or implicit). We could potentially look at any of the information we have on a course.

Units

Grade Type (Letter Grade or Pass / Fail)

Term

Lecture Hours

Practice Hours

Extra Hours

Language

Lecturers

Responsible Unit

Course Areas

Description

Learning Goal

Obligatory Acitvities

Prerequisites

Preexisting Knowledge

Course Material

Unit Reduction

Exam (type, percentage of grade, aids and date)

In order to figure out what a user prefers we looked at all of the courses he/she had rated higher than or equal to his/her own average rating. If we had only looked at courses that had a set range of ratings, say higher than 3 out of 5, we would not take into consideration the fact that users use the rating scale differently. Some users give mediocre scores to almost all of their courses, while others give extremely high and low scores to all the courses they rate. After we have found the favorite courses for a user, we then extract the features we are interested in from the courses. Initially we decided to use "Course Areas" and "Lecturers". Course area was selected because it is a good indication of the types of courses a user prefers. Lecturer was chosen because studies have shown that a good lecturer is key to how a student feels about a course [Marsh, 1984].

After the initial user survey we added a new CB type, courses that are popular with older students in the same degree program. At NTNU many students need to take courses that are not from their own degree program. Finding a good course from another degree program can be challenging, by suggesting the courses that are the most popular with older students we hope to alleviate this problem. In this recommendation type we only show courses that are not obligatory for the degree program of the user.

After we have found the features that the user has shown a preference for, we then find all of the courses that have those features. We then sort the courses in each feature-category by the total rating they have on Classmate, and present the highest ranked courses in each feature-category as the recommendations.

4.3 Hybrid

In the first version of the recommender system (the one used in the initial evaluation stage) we presented all of the recommendations in one big list. In order to better explain the recommendations we decided to show the results in groups in the second version. Each category has a header that specifies why the courses were recommended (e.g. "Courses with a professor you seem to like"). The categories are sorted by how well people liked the results from that category during the evaluation. We do not however attempt to guarantee any sort of sorting inside each of the categories, some of the categories are sorted (collaborative filtering for instance), while in others we sort by total rating. Since the sorting is not consistent between the categories, and the total number of recommendations is kept low, we believe telling the user that the courses are sorted in order of relevance would not improve the experience.

4.4 Filtering

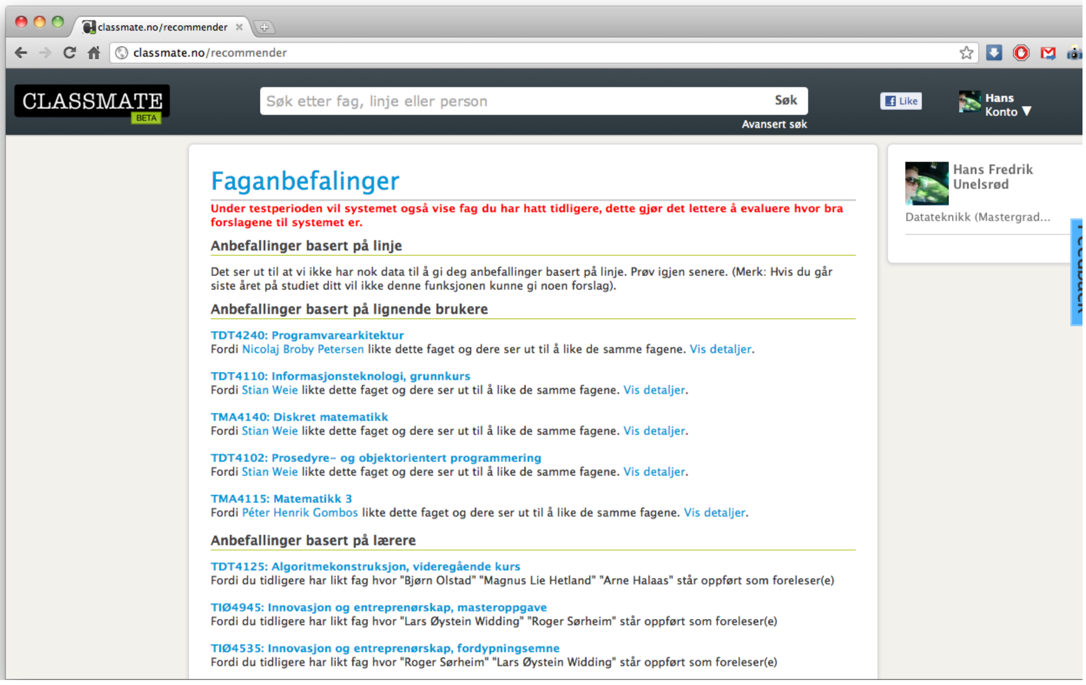
Classmate.no already has "Advanced Search" functionality where users can use filters to find the type of courses they want. Examples of filters include number of units, lecturer, exam type and the name of the course. Students often need to find a course that fits a set of requirements, for instance a course with 10 units. And some students have preferences that might not be detected by the recommender system, for instance courses without any required work except for the exam. In these cases using advanced search to filter out courses that are not relevant first is a good way of getting better recommendations. After the user has filtered out the courses he/she does not want, he/she can then click a link to the recommender system that then gives them recommendations inside the results from their filtering. We disabled this feature during the user evaluation in order to make the recommendations more standardized.

4.5 Presentation

We decided early on in the project to not focus on presentation. That is why we went with a simple textual list of the recommendations. Other options include more graphical presentations like for instance a graph of the recommendations and how they relate to the user‘s degree program. As mentioned we experimented with both a single list and multiple lists grouped by recommendation-reason, but in the end the users seemed to prefer grouped lists so that is what we use in the final implementation.

4.6 Explanation

Each category of recommendations has its own type of explanation. The collaborative filtering recommendations show the name of the user that links the user getting the recommendation to the course being recommended, we also show the courses that link them together. For the content based recommendations we show the reason for a course being recommended in plaintext, for example if a course is recommended because the user has shown a preference for courses with "Person X" as lecturer, we say "Course Y is recommended because you have shown a preference for courses lectured by Person X"



Faganbefalinger

Under testperioden vil systemet også vise fag du har hatt tidligere, dette gjør det lettere å evaluere hvor bra forslagene til systemet er.

Anbefallinger basert på linje

Det ser ut til at vi ikke har nok data til å gi deg anbefallinger basert på linje. Prøv igjen senere. (Merk: Hvis du går siste året på studiet ditt vil ikke denne funksjonen kunne gi noen forslag).

Anbefallinger basert på lignende brukere

TDT4240: Programvarearkitektur
Fordi Nicolaj Broby Petersen likte dette faget og dere ser ut til å like de samme fagene. Vis detaljer.

TDT4110: Informasjonsteknologi, grunnkurs
Fordi Stian Weie likte dette faget og dere ser ut til å like de samme fagene. Vis detaljer.

TMA4140: Diskret matematikk
Fordi Stian Weie likte dette faget og dere ser ut til å like de samme fagene. Vis detaljer.

TDT4102: Prosedyre- og objektorientert programmering
Fordi Stian Weie likte dette faget og dere ser ut til å like de samme fagene. Vis detaljer.

TMA4115: Matematikk 3
Fordi Péter Henrik Gombos likte dette faget og dere ser ut til å like de samme fagene. Vis detaljer.

Anbefallinger basert på lærere

TDT4125: Algoritmekonstruksjon, videregående kurs
Fordi du tidligere har likt fag hvor "Bjørn Olstad" "Magnus Lie Hetland" "Arne Halaas" står oppført som foreleser(e)

TIØ4945: Innovasjon og entreprenørskap, masteroppgave
Fordi du tidligere har likt fag hvor "Lars Øystein Widding" "Roger Sørheim" står oppført som foreleser(e)

TIØ4535: Innovasjon og entreprenørskap, fordypningsemne
Fordi du tidligere har likt fag hvor "Roger Sørheim" "Lars Øystein Widding" står oppført som foreleser(e)

Hans Fredrik Unelsrød
Datateknikk (Mastergrad...

Figure 4.1: Screenshot of the course recommender system

Chapter 5

Evaluation

Our evaluation included both user trials with questionnaires and synthetic tests. We started running trials with users as soon as we had a functional recommender system. We wanted to start doing evaluations as early as possible in order to help us figure out with signals to focus on when developing the content-based part of the recommender. We also wanted to get feedback on how well the collaborative filtering part was working. We also wanted to confirm our assumptions regarding what students at NTNU look for when selection a course.

5.1 Initial User Survey

Our main goals with the initial user trial was to figure out if people feel like they need help in finding courses, how good the perceived quality of the initial results were, whether or not the assumption that social relations play a big role in course selection and figure out what the most important factors are when a student selects a course. With these goals in mind we set out to write a questionnaire that would help us get an answer to our questions [Appendix A]. We used Google Forms to create and distribute the questionnaire, the survey was sent out to 50 randomly chosen Classmate.no users. Unfortunately we only got a response from 7 out of the 50 users we sent the survey out to (14 % response rate). This equates to a confidence interval of 36 with a 95 % confidence level (using a population size of 500, the number of registered Classmate users at the time of doing this survey). Even though we were hoping for a better response rate we still managed to get some useful data from the initial survey.

Since the CB part of the recommender could not be evaluated in a good manner using a synthetic test we decided to apply all of the CB signals that seemed logical and that gave us good results during initial testing. We then evaluated them by letting users give feedback on the results during the user evaluations. The result of the first user evaluation was that we removed one of the signals, course area, since most users reported that the recommendations from this signal was not interesting. After investigation we discovered that most users got the same recommendations from this signal. Most students at NTNU have taken several courses in the course area called "Technical Courses", and the majority of the users had their most highly rated courses in this course area, this resulted in most users being recommended the same "Technical Courses". Another change we made after the initial evaluation was to split the results up into groups/categories, this was done to make it easier for users to understand why a given course was recommended.

In addition to the changes outlined in the previous paragraph, we also improved explanations of the CF recommendations by allowing users to view which courses they had in common with the user that caused a course to be recommended. When we did the initial user survey we were not yet finished with evaluating the different correlation algorithms. So in the initial version of the recommender system we used our own distance based metric. But the test we later ran showed that Pearson Correlation performed the best on our data set, therefore we changed to Pearson Correlation in the final version of the recommender system.

5.2 Final User Survey

5.2.1 Goal

In the final user evaluation our goal was to find out how well our recommender system performed with as many users as possible. In order to do this we did another user survey [Appendix B]. Since the goal of this survey was to gather as much data as possible from as many users as possible we tried to streamline the questionnaire. We did this by focusing on questions with quantitative answers, questions which are quick and easy for a user to respond to. Another advantage with quantitative questions is that it leaves less room for interpretation from us.

At the time of doing the survey Classmate had roughly 1.000 users. When doing the first survey most of the users where early adopters who where more likely to be willing to try out and give feedback on a new system like this. So we were expecting the response rate to be even lower for this survey. We wanted to at least achieve a confidence interval lower than 10. In order to get that using a 95 % confidence level we would have to get 88 people to fill out our survey.

5.2.2 Method

We sent out an email to all of the students at the Department of Computer Science at NTNU, roughly 600 people. Since we had to assume that almost all of the recipients of the email where not existing Classmate users, we included instructions on how to register an account on Classmate. Unfortunately we also needed all of the participants to go in and add all of their previous courses and give a rating to as many of them as possible. As mentioned we view the integration of this system into Classmate as one of its greatest strengths. But for users who have to go through and rate all of their courses just to get to the recommender system, the benefits of this is all but lost. Another hassle for the unregistered users is that our recommender system runs part of its collaborative filtering in an offline process, this means that new users will not get recommendations based on collaborative filtering until after they have waited 24 hours after rating their courses. Even though we knew some people would be put off by the amount of work they had to do before being able to answer the survey, we where not expecting the number of responders to be as low as it was. Only 10 people had answered the survey three days after we sent out the email.

Originally we had hoped that we would get enough answers from the email we sent out to students at the Department of Computer Science. But since we realized that we needed to get more users to evaluate the system in order to get any useful data, we decided to send out an email to some of the Classmate users and hope that they would be more willing to evaluate the system and answer out survey. So we sent out an email to all of the users of Classmate. We where oping that since they did not have to go through the process of registering and give ratings to their courses, that we would get a higher response rate from these users. After three days around 200 people had tried out the recommender system, but only 33 people in total had answered the survey. Even though the number of responses where lower than we had hoped, we still managed to get a confidence interval of roughly 17 (again using a 95 % confidence level).

One positive thing about the users who responded is that they represent a fairly good variation in degree subjects, see Table 5.2.2, and class years, see Table 5.2.2.

Degree Subject	Percentage of Respondents
Compuer Science	22 %
Economy	3 %
Chemistry	3 %
Construction	9 %
Entrepreneurship	9 %
Other	47 %

Table 5.1: The degree subjects of the users who completed the survey

Class Year	Percentage of Respondents
1	6 %
2	25 %
3	9 %
4	31 %
5	28 %

Table 5.2: Class year of respondents

If we had simply asked users wether or not they liked the suggestions they where given, we would have to trust users to make a judgement on wether or not a course would fit them based on only the name of the course. The recommender system does not recommend courses the user has already rated or added to his course history, therefore all of the courses rated would be fairly unfamiliar to the user. In order to solve this we temporarily made the recommender system include courses that a user had rated or added to his course history. That way users where better able to judge the quality of the recommendations. If all of the courses recommended where courses that user had liked in the past we would take that as an indication that future recommendations would also be good.

5.2.3 Results

We will look at the results in sections, one for each of the different types of reasons we have for recommending a course. In each section we asked the users how pleased they where with the recommendations, and asked them to rate the recommendations on a scale from 1 to 5. Unfortunately the results where evenly distributed across the possible ratings in most of the sections. And since the confidence interval is low, we decided to make the results less fine grained by grouping the ratings into "Bad", "OK" and "Good". With "Bad" being a rating of 2 or lower, "OK" being a rating of 3 and "Good" being a rating of 4 or higher. One other thing to note is that not all of the sections sum up to 100 %, this is because all of the percentages are of the total number of responders, and not all responders answered every question.

The first section is courses recommended based on other users with similar ratings, collaborative filtering, see Table 5.2.3. It is clear to us that most users where not as pleased with these recommendations as we had hoped. This could be a sign that simple collaborative filtering (even with the additional signals we included) is not a good fit for this data-set. But we would need to do additional evaluation to find out exactly why the results where as low as they were. It could be that a larger data-set or more tuning of the collaborative filtering weights would have lead to better results.

Rating of Results	Percentage of Responses
Bad	44 %
OK	22 %
Good	34 %

Table 5.3: Rating of recommendations from collaborative filtering

The second is the recommendations based on which courses users in the same degree subject usually take in later class years, see Table 5.2.3. Here the results clearly show that most users were not satisfied with the recommendations they where given, but there is a possible correlation between the number of users in the 5th class year and the number of users who where dissatisfied with these recommendations. Users who are in their last year of their study can of course not get this type of recommendations. In hindsight we realize that the survey should have stated more clearly that users who did not received recommendations of this type should not answer this question.

Rating of Results	Percentage of Responses
Bad	48 %
OK	21 %
Good	3 %

Table 5.4: Rating of recommendations from degree subject courses

The third section is the recommendations based on which lecturers a user has shown a preference for in the past, see Table 5.2.3. It is clear that the quality of this section of recommendations vary a lot from user to user. This makes sense since Classmate, at the time of writing, only had support for rating the quality of the lectures in a course as a whole, regardless of which lecturer that user had when they took the course. Apposed to rating each lecturer individually, which is something Classmate now supports, and which should be integrated into the recommender system in the future.

Rating of Results	Percentage of Responses
Bad	44 %
OK	6 %
Good	37 %

Table 5.5: Rating of recommendations from lecturer

The final section of results is the total satisfaction with the recommendations, see Table 5.2.3. 40% of responders rated their total satisfaction 4 or higher. This is definitely lower than we had expected and leads us to believe that the quality of the recommendations is not as high as it should be in order for the system to be a success. We would however have to do more analysis and further evaluations in order to find out why.

Rating of Results	Percentage of Responses
Bad	37 %
OK	22 %
Good	40 %

Table 5.6: Total satisfaction with recommendations

Chapter 6

Conclusion

6.1 Summary

In this thesis we have constructed a recommender system for course selection. We started out by doing a survey of the literature on recommender systems for course selection, then we defined the goals we wanted to achieve in Section 1.2.

6.1.1 Goal 1

Our primary goal was to develop and test a recommender system that uses a hybrid of collaborative and content filtering methods to give course recommendations. We have achieved this by using memory based collaborative filtering with k-nearest neighbor search to find closely correlated users, and content filtering based on the most popular courses with older students and which teachers the user has shown a preference for.

6.1.2 Goal 2

Our second goal was to integrate the social graph and the user‘s chosen degree subject as signals in our collaborative filtering process. We did this by testing the collaborative filtering system using precision and recall in order to find the best correlation algorithm and the optimal weighting of the additional signals. The results of these tests where that we ended up not utilizing the social graph, since it did not improve accuracy. The chosen degree subject however turned out to provide a good boost in accuracy.

6.1.3 Goal 3

Our third and final goal was to develop a recommender system that can be used in a production environment. We do not feel that we have been able to achieve this goal to as high a degree as we had hoped. The system currently does all of the correlation calculations in an off-line process that runs every 24 hours. This means that the every time a user rates a course, they have to wait 24 hours before that rating will be reflected in the recommendations. The process of getting the recommendations however is done in real-time.

Our user evaluation showed that the quality of the results is not consistent across users, and the total level of satisfaction with the system is lower than we had hoped. Only 40 % of users would rate the system 4 out of 5 or higher. This is definitely lower than we would like to see in order to call the system a success with the users of Classmate.

6.1.4 Research Question 1

The first part of this research question was "will building on something like Classmate.no be enough to alleviate the problem of having to force users to fill out long questionnaires before they can get recommendations?". The answer to this is really two-fold. For users who are already registered and have rated a few of their courses the answer is yes, they do not have to do any extra work in order to get recommendations. For users who are either not registered or are registered, but have not rated any of their courses yet, the answer is no. They still have to enter quite a lot of information manually. Classmate does make this easier by automatically knowing which courses the user is likely to have taken based on their chosen degree subject and class year. But the user still has to manually rate some of their courses. Hopefully this is something that most users have an inclination to do on their own, regardless of the recommender system. For those people the recommender system can be hidden until they have entered enough data, in that sense the user is not likely to feel like they had to do a any work in order to get their recommendations.

The second part of this question was how building upon an existing and relatively (in comparison to previous research) large dataset would impact the development of the recommender system. This question is difficult to answer, since we have not done any attempts at recreating any of the experiments done in other papers we can not make any direct comparisons. We can however with a large degree of certainty say that having a huge dataset helped us in testing out the additional collaborative filtering signals as well as choosing the appropriate correlation algorithm. If we where starting from scratch the tests we did of the collaborative filtering process would have to be done with just the data we gathered in the initial user survey. And seeing as only 7 people completed that survey, we would have a really limited dataset to work with. Another option would have been to try to get the same amount of data from the school, albeit a slightly different type of data. This could have worked, but we would not have gotten the honest personal opinion from the students.

We feel that building on Classmate was a great advantage in making the recommender system easier to use for the users and in making the system better by allowing us to do extensive synthetic testing.

6.1.5 Research Question 2

Research question 2 was "How well will including additional signals into the collaborative filtering process work?". As we mentioned in Section 6.1.2, using friend relationships to weight the correlation used in the collaborative filtering process did not turn out to improve the accuracy of the recommendations. The user‘s chosen degree subject however was shown to lead to an improvement in precision of almost 8%. In hindsight we wish had taken the time to test even more collaborative filtering signals, we believe that there might be potential for additional gains in accuracy by using for example class year in combination with degree subject.

6.1.6 Research Question 3

Research question 3 was "How will students respond to using a recommender system for course selection?". We did not focus extensively on this question during our evaluations. But what we can say is that 79 % of the responders said that they where pleased with the explanation given as to why a specific course was recommended. And 67 % of responders said that they would consider choosing a course based on a recommendation from a system like this. These answers lead us to say that it looks like students (at least at NTNU) would like to have a system like this available, and that theses students would be likely to use the system when choosing courses. It is likely that most students would do additional research on their own before choosing a course. This is another area where being integrated into Classmate is an advantage, students can quickly and easily click a link to the course page on Classmate for the course they are interested in and see reviews and ratings from other students that have taken the course.

6.2 Discussion

The lack of users who participated in our user survey is a clear threat to the validity of the results. It is also clear to us that we should have asked the users in the final survey for their username (something we did in the initial survey), so that we could have looked at the recommendations they got. This could have enabled us to do a more thorough analysis of why some users where really happy with their recommendations, while others felt the opposite.

Building on top of Classmate did not help us to the degree we where expecting. We had expected more users to want to use the system, and we expected the fact that we started out with a fairly large data-set to help out more in the performance of the collaborative filtering recommendations.

Even though we were not able to come to a clear conclusion on how good the recommender system will perform in real life, we were able to show that adding additional signals to the collaborative filtering process is a good fit for course recommender systems.

We plan to take the lessons learned and some of the technology we have developed while doing this thesis, and convert it into a permanent feature on Classmate.

6.3 Contributions

The main contribution that we think this thesis makes is the concept of course recommendation using collaborative filtering with weighting of users. Our initial synthetic testing shows that this technique has merit and should be investigated further. We also think that the idea of basing a course recommender system on an existing website for rating of courses is a novel idea, and one that could prove to be necessary in order to make a recommender system that is able to easily scale to many schools.

Chapter 7

Future Work

Our primary goal for future work is to do a more extensive evaluation of the system we have developed. This would also include implementing refinements based on the new evaluation. Specifically we would like to focus on finding more signals to integrate into the collaborative filtering process.

We would also be interested in adding more signals to both the collaborative filtering and the content filtering parts of the system. One signal in particular that we would be interested in is grades.

Another aspect that we would like to look into is support for finding courses that the users has the correct prerequisites to take, and how choosing one course will affect the user‘s options in the future. A related aspect is finding courses that do not have conflicting schedules.

And finally we would like to improve the performance of the system, especially the calculation of correlation between users. Doing the calculations in a progressive manner by looking at only the new ratings that have been posted since the last time, instead of going through all users every time. Should lead to a drastic improvement in performance, and could make the entire system run in basically real-time. Scaling is a related issue that would also be fascinating to look further into.

Bibliography

Adomavicius, G. and Tuzhilin, A. (2005). Personalization technologies: a process-oriented perspective. *Communications of the ACM*, 48:83–90.

Balabanovic, M. and Shoham, Y. (1997). Fab: Content-based, collaborative recommendation. *Communications of the ACM*, 40:66–72.

Bilgic, M. (2005). Explaining recommendations: Satisfaction vs. promotion. In *In Proceedings of Beyond Personalization 2005, the Workshop on the Next Stage of Recommender Systems Research(IUI2005*, pages 13–18.

Breese, J. S., Heckerman, D., and Kadie, C. (1998). Empirical analysis of predictive algorithms for collaborative filtering. pages 43–52. Morgan Kaufmann.

Burke, R. (2002). Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12:331–370.

Cesar Vivalardi, Javier Bravo, L. S. A. O. (2009). Recommendation in higher education using data mining techniques. Master's thesis, Universidad Autonoma de Madrid.

Cleverdon, C. W., Mills, J., and Keen, M. (1966). Factors determining the performance of indexing systems.

Farzan, R. and Brusilovsky, P. (2006). P.: Social navigation support in a course recommendation system. In *In proceedings of 4th International Conference on Adaptive Hypermedia and Adaptive Web-based Systems*, pages 91–100. Springer Verlag.

Goldberg, D., Nichols, D., Oki, B. M., and Terry, D. (1992). Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35:61–70.

Herlocker, J. L., Konstan, J. A., Terveen, L. G., John, and Riedl, T. (2004). Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems*, 22:5–53.

KoKang Chu, Maiga Chang, Y.-T. H. (2009). Desining a course recommendation system on web based on the students' course selection records. Master's thesis, Chung-Yuan Christian University.

Lillegraven, T. N. and Wolden, A. C. (2010). Design of a bayesian recommender system for tourists presenting a solution to the cold-start user problem.

MacManus, R. (2009). A guide to recommender systems.

Makhoul, J., Kubala, F., Schwartz, R., and Weischedel, R. (1999). Performance measures for information extraction. In *In Proceedings of DARPA Broadcast News Workshop*, pages 249–252.

Marsh, H. W. (1984). Students' evaluations of university teaching: dimensionality, reliability, validity, potential biases and utility (1984) students' evaluations of university teaching: dimensionality, reliability, validity, potential biases and utility (1984) students' evaluations of university teaching: Dimensionality, reliability, validity, potential biases and usefulness. Master's thesis.

NTNU (2011). Ntnu course list.

O'Mahony, M. P. and Smyth, B. (2007). A recommender system for on-line course enrolment: an initial study.

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., and Riedl, J. (1994). Grouplens: An open architecture for collaborative filtering of netnews. pages 175–186. ACM Press.

Rodgers, J. L. and Nicewander, W. A. (1988). Thirteen ways to look at the correlation coefficient. *The American Statistician*, 42:59–66.

Salton, G. (1989). *Automatic text processing: the transformation, analysis, and retrieval of information by computer.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Sarwar, B., Karypis, G., Konstan, J., and Reidl, J. (2001). Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, WWW '01, pages 285–295, New York, NY, USA. ACM.

Shardanand, U. and Maes, P. (1995). Social information filtering: Algorithms for automating "word of mouth". pages 210–217. ACM Press.

Sørmo, F., Cassens, J., and Aamodt, A. (2005). Explanation in case-based reasoning—perspectives and goals. *Artif. Intell. Rev.*, 24:109–143.

Symeonidis, P., Nanopoulos, A., and Manolopoulos, Y. (2008). Providing justifications in recommender systems. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 38(6):1262 –1272.

Tintarev, N. and Masthoff, J. (2007). A survey of explanations in recommender systems. In *Data Engineering Workshop, 2007 IEEE 23rd International Conference on*, pages 801 –810.

van Rijsbergen, C. J. (1979). *Information Retrieval.* Butterworths, London, 2 edition.

Weber, R., Schek, H.-J., and Blott, S. (1998). A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces. In *Proceedings of the 24rd International Conference on Very Large Data Bases*, VLDB '98, pages 194–205, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Young, J. R. (2011). The netflix effect: When software suggests students' courses.

Appendices

7.1 Appendix A

Questionnaire 1
Info
Name
@stud e-mail
Degree Subject
School yearår

Quality
How well would you rate the suggestions from the "ser ut til å like lignende fag" section?
How well would you rate the suggestions from the "fordi du tidligere har likt fag av typen X" section?
How well would you rate the suggestions from the "ser ut til å like foreleser X" section?

What was the best recommendation?
What was the worst recommendation?

Would you have preffered simple filters that showed the total rating for that course, but did not do any automated recommendations?
If yes to the preivous question, why?

Improvement
Which of these factors do you focus mostly on when selecting a course?

- Lecturer, Curriculum, Subject, Tasks, Exam

Other factors

Whats the biggest problem you have when trying to find a course today?
Have you ever selected a course simply because your friends where taking it?

7.2 Appendix B

Questionnaire 2
Info
Degree Subject

School year

Quality
How well would you rate the suggestions from the ”ser ut til å like lignende fag” section?
How well would you rate the suggestions from the ”ser ut til å like foreleser X” section?
How well would you rate the suggestions from the ”Fordi dette er et av de mest popul$\frac{3}{4}$re og blant de best likte ikke-obligatoriske fagene som tas på din linje” section?

Total
How well would you rate the recommendations?
Where you convinced that the recommendations where good suggestions based on the explanation given?