A STUDENT PROGRAM RECOMMENDATION SYSTEM PROTOTYPE

Queen Esther Booker, Minnesota State University, Mankato, queen.booker@mnsu.edu

ABSTRACT

Recommendation systems have emerged as a useful e-commerce tool to assist customers in making purchases based on similarities and preferences of others. However, their potential has not been highly utilized to help in one area that could assist both professors and students - student advising. Using a recommender system has potential risks such as lower student retention if the system makes poor recommendations but it also has the potential of helping to guide potential students to a wider range of majors and courses that they may not have otherwise considered. This paper discusses a prototype recommender system for advising pre-applicants on probable courses of study at a university.

Keywords: Interactive systems, Student Advising, Recommendation Systems

INTRODUCTION

One challenge the prospective student faces when applying to a college is "can it meet my educational needs?" Oftentimes it is simply a guess. Students consult current enrollees if possible, they may attend an informational session on about the college, or they search online to find the college's website. But even when the prospective student finds the website, they have to try to figure out exactly what they want to know. So, most students actually go to their selected colleges *hoping* to get the academic experience they imagine. When the school does not meet expectations or the student is unable to achieve the academic levels necessary to remain enrolled, the student is not retained. Higher education institutions have a significant amount of data intelligence that could potentially assist students in making a better decision about the university and the potential programs. They have the academic records of existing students, data on professors, course descriptions and prerequisites, and programs of study. Yet, little has been done with sharing that intelligence with prospective students.

In 200The intelligence about the university is held across many divisions within the university but when combined can provide prospective students with a rich report that is more informative than having the student review a course catalog or bulletin. Being able to aggregate the information about the university in such a way as to not violate privacy issues could benefit students as consumers as a way to effectively aid in the higher education selection process [1].

Such a system could act as a screening process, helping students get a snapshot of programs of study that may interest them, and reduces the possibility of information overload [2]. Without such screening, students may stop searching long before exhausting the set of relevant courses [3]. Such a program could potentially reduce the likelihood of the student not completing an academic program, decreasing the already statistically high attrition rates that currently plague enrollments at several colleges and universities.

To overcome the inherent difficulty of product selection in the e-commerce market, smart recommendation agents have emerged as an effective tool in product screening and filtering [4, 5, 6, 7, 8]. Smart agents counteract the information explosion problem by "considering" a large number of alternatives and recommending only a small number to the consumer, similar to getting word of mouth recommendations from hundreds of people who share similar tastes and experiences rather than one or two people whose opinions a person trusts but may not share in similarity in tastes and experiences. An effective screening system that is highly correlated with consumers' overall preferences will have utility almost as high as if every alternative in the universe were inspected— but with dramatically reduced effort on their part [9]. A search algorithm that provides such a service to students could assist in select courses and perhaps a course of study that are consistent with their interests and their abilities.

Currently there are many recommendation systems online such as match-making companies, online college selection programs, movie recommendations, music recommendation and even clothing accessories. Many of these recommendation systems are mono product oriented and simply rely on past behaviors of customers such as the recommendation for the purchase of multimedia and books such as Amazon.com.

Most recommendation systems use some form of clustering algorithm to recommend several items that

others who have purchased items similar to the customer's had recently purchased. In this case, the system must recommend possible several majors plus several general education courses to meet the needs of the students. As this is an early prototype, the current system uses clustering as the primary algorithm to simply examine the potential of such a program.

This paper discusses the prototype system including its design, current experiments and areas for future study. The research is based on on-going experiments to improve the use of data intelligence to provide better support for the modern student.

RECOMMENDER SYSTEMS AND UNDERLYING ALGORITHMS

Recommender systems are information systems that provide suggestions to consumers that most likely meet their needs and preferences. The systems use as a base a large amount of data collected over time and from other consumers, and uses any number of techniques to provide suggestions to the consumer, usually for purchase. All recommendation techniques have strengths and weaknesses but have a common purpose of providing the best recommendations that will lead to a successful purchase by the consumer. A variety of techniques have been use for performing recommendation, including content-based, collaborative, and knowledge-based and other techniques.

Content-based systems use information filtering for their recommendations [10]. These systems use associated features such as the same book or artist as the objects of interest. For example, text recommendation systems like the newsgroup filtering system NewsWeeder [11] uses the words of their texts as features. A content-based recommender learns a profile of the user's interests based on the features present in objects the user has rated. Schafer, Konstan & Riedl call this "item-to-item correlation." The type of user profile derived by a content-based recommender depends on the learning method employed. Decision trees, neural nets, and vector-based representations have all been used. As in the collaborative case, content-based user profiles are long-term models and updated as more evidence about user preferences is observed. [12]

Collaborative filtering (CF) is the most common recommendation technique to date. The basic idea of CF-based algorithms is to provide item recommendations or predictions based on the opinions of other like-minded users. The opinions of users can be obtained explicitly from the users or by using some implicit measures. The goal of a collaborative filtering algorithm is to suggest new items or to predict the utility of a certain item for a particular user based on the user's previous likings and the opinions of other like-minded users. In a typical CF scenario, there is a list of m users $U = \{u_1, u_2, \ldots u_m\}$ a list of n items $I = \{i_1, i_2, \ldots, i_n\}$. Each user has a list of items within the list, about which the user has expressed his/her opinions. Opinions can be explicitly given by the user as a rating score, generally within a certain numerical scale. There exists a distinguished user u_a called the active user for whom the task of a collaborative filtering algorithm is to find an item likeliness that can be of two forms: prediction or recommendation. [13]

Knowledge-based recommendation attempts to suggest objects based on inferences about a user's needs and preferences. In some sense, all recommendation techniques could be described as doing some kind of inference. Knowledge-based approaches are distinguished in that they have functional knowledge: they have knowledge about how a particular item meets a particular user need, and can therefore reason about the relationship between a need and a possible recommendation. The user profile can be any knowledge structure that supports this inference. In the simplest case, as in Google, it may simply be the query that the user has formulated. In others, it may be a more detailed representation of the user's needs [12, 14]

THE SYSTEM DESIGN

The purpose of this on-going research is to determine if it is possible to design a system that could essentially perform a pre-selection program plan for prospective students that would give them a potential road-plan for an academic career at Minnesota State University, Mankato. This study grew from inquiries from students who learned about the Management Information Systems (MIS) major too late in their academic careers to switch into the program, and from students who had taken the wrong pre-requisites to enroll in the MIS upper division courses and thus resorted to another major. From these inquiries, it has become clear that helping students identify a program that matches their interests earlier in their academic careers could potentially reduce attrition, increase student satisfaction and improve recruitment for the university. Two previous prototypes have been developed to provide some support for students. The first was a system developed to mimic the word of mouth approach of students recommending courses

to other students. [15] This system only used courses students had previously taken and student ratings to match user preferences. The second was an experimental system designed to recommendation system designed to identify probably courses based on courses the student had previously taken using linear classifiers as the underlying algorithm. [16] This current prototype design is modest just to test the feasibility of the concept of expanding to program identification as well as course suggestions, and using the college data as well as data about students. However, this prototype relies less on what students recommend for courses and more on how students rate the course.

The initial design includes actual data from the current course bulletin, the registration data for the past five years to identify patterns of course offerings versus what is in the course bulletins, and faculty evaluation and rank. The last data set was fabricated student data of students who had enrolled in each of the courses, their high school and current college GPAs, their majors and their ranks such as freshman, sophomore, junior and senior.

The previously described five data sets are the foundations of the system. The user interface is an input screen where students could enter their academic interests, high school GPA and ACT/SAT scores as shown in Figure 1.

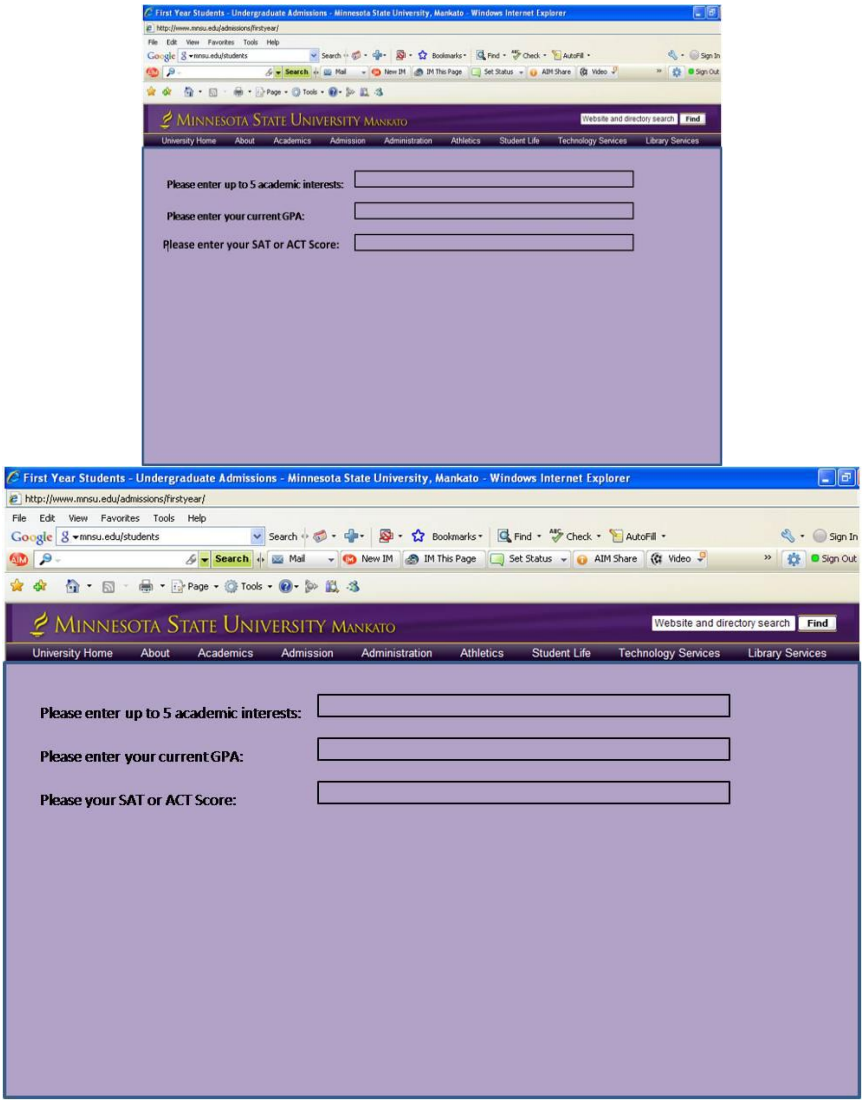

Figure 1. User Data Entry Screen

The data processing begins by cleansing the data by checking for typographical errors in the academic interests and correcting them. For example, if science is spelled "sceince" then it is corrected to "science." The backbone for the cleansing process is a simple soundex algorithm. Then the GPA score is checked for validity. If it is not between 0 and 4, the student is informed that a grade point average is typically between 0 and 4 and is asked to correct the GPA before continuing. If the student's GPA is not between 0 and 4 then the user is informed that the system cannot process the request at this time. If the GPA is blank it is acceptable but a negative GPA or a GPA above 4 is currently not acceptable.

The next process generates n! academic itemsets based on the number of inputs in the academic interests box. It then generates a performance itemset of the high school GPA and standardized test scores. The next process uses a basic *k-means* clustering algorithm to match and rank the performance itemset with current students in the system. Using the matches, the performance matches are then matched against the n! academic interests and ranks potential majors based on the number of similar matches. For example, if a student inputs computers, art, science and math, one cannot assume that computers are the primary interest or that math is the least of the interests. So all possible items are analyzed and majors stored in a temporary bin to match against majors for students with similar interests and performance itemsets. All of the possible matches with a rank of 50% or higher are returned to the student in a display similar to the one in Figure 2.

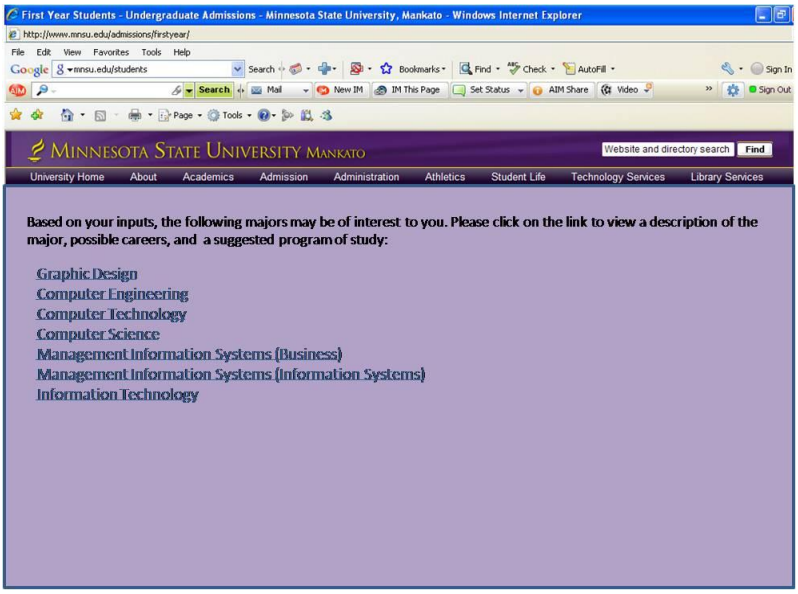

Figure 2. Display of Major Results

Each returned major is a hyperlink to a just-in-time document that is generated from the databases. The document contains the current course bulletin description of the major, the college and/or the department under which the major falls, and a recommended program of study as shown in Figure 3.

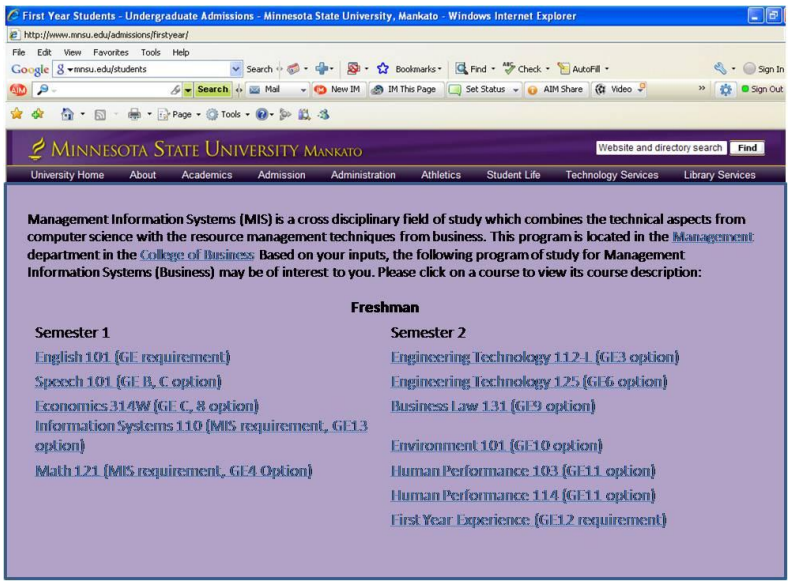

Figure 3. Suggested Program of Study

The courses are pulled from the course bulletin in the following order:

1. Required courses or prerequisites for the major that also fulfill general education requirements
2. Required courses or prerequisites for the major that do not fulfill general education requirements
3. Optional courses to fulfill general education requirements that match at least one academic interest AND the performance itemset within parameters. If the performance itemset does not generate a match then the algorithm searches on major and current GPA to find those courses in which students in similar majors took and did well.
4. Optional courses to fulfill the major requirements, using a similar algorithm to that in Step 3.

Once all the potential courses are selected, those that are requirements are automatically placed in the schedule for the appropriate time to be taken. Then optional courses that meet the same requirement either the major or the general education courses are added based on rankings. The ones with the highest matches to both the performance and interests are added first until all the degree requirements have been completed. The resulting display is a *suggested* program of study. Students can then select any of the courses and read the course description, a list of the most recent instructors and their ranks, and their overall evaluations as shown in Figure 4.

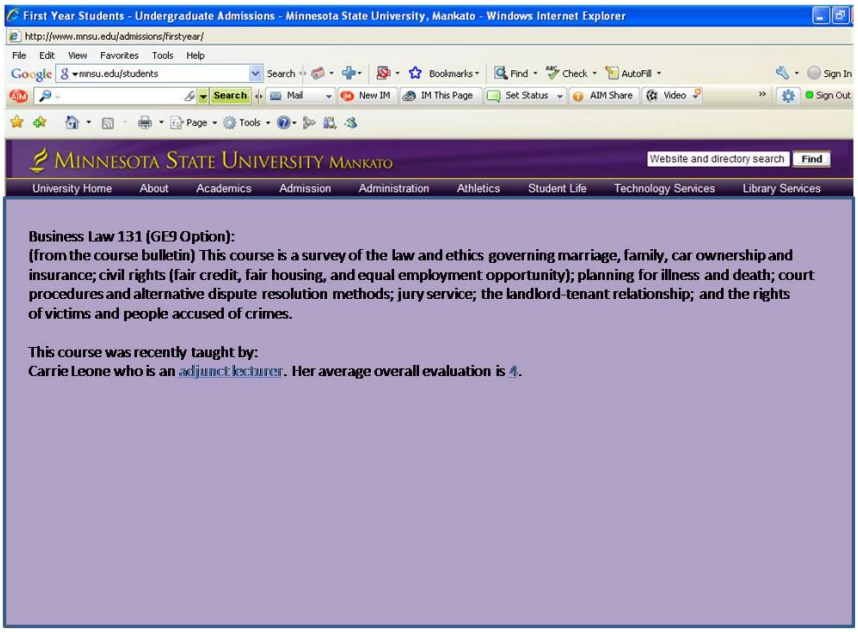

Figure 4. Sample course description

EXPERIMENTS

Because the system is a prototype, it has not been tested on prospective students. Instead, the student information database was populated with fabricated data so that the results for specific inquiries would be known. There were two major concerns for the systems were the accuracy of the results and the speed of processing. The itemset matching and ranking were occurring in real time so the system was tested for performance based on the number of interests the user entered and the number of users accessing the system simultaneously using a Dell PowerEdge R900 server. The system also used copies of the course catalogue, instructor data, etc to minimize the experiments impact on the actual systems data servers.

The server was tested first with for one user with between one and twenty interests. Recall the base algorithm for the recommendations was clustering, specifically k-means clustering. The results of the k-means clustering were compared to a random selection of courses based on requirements for degree completion. The results of matching accuracy, defined as finding the right majors based on matching the performance itemsets with the academic itemsets and then the appropriately matched courses. As the number of academic interests increased above 12, the clustering algorithm returned the same results as a random selection indicating that the k-means approach only works as well as a random selection at that level. Further, the algorithm returned decreasingly relevant results starting with two interests with the steepest decline beginning with six academic interests. The overall performance results can be found in Table 1.

Table 1. Academic Interest Itemset Matching Results

Number of Academic Interests	Number of Itemsets to Match	Random Course Selection Matches (non requirements)	Cluster Course Selection Matches	Processing time (in seconds)
1	1	40%	95%	0.001

2	2	41%	92%	0.001
3	6	40%	89%	0.001
4	24	40%	89%	0.002
5	120	40%	87%	0.003
6	720	43%	85%	0.05
7	5040	43%	80%	0.1
8	40320	39%	76%	0.1
9	362880	38%	72%	1.2
10	3628800	38%	70%	1.5
11	39916800	43%	55%	1.7
12	479001600	40%	41%	1.9
13	6227020800	43%	40%	4.1
14	87178291200	39%	40%	4.6
15	1.30767E+12	40%	41%	4.9
16	2.09228E+13	40%	40%	5.1
17	3.55687E+14	41%	40%	7.3
18	6.40237E+15	42%	41%	7.6
19	1.21645E+17	39%	40%	7.9
20	2.4329E+18	42%	41%	8.4

CONCLUSIONS AND NEXT STEPS

The program and course recommendation system has potential. Beyond the recommendations to potential students, there are other ways in which such a system can be appealing based on the student's needs. For example, the application could be used to help enrolled students manage their academic careers, such as moving courses from one semester to another when something such as a closed class might keep them from enrolling in the planned program. Students could also perform what if analysis such as what if the student wanted to attend summer school or double major. The largest benefit though, is the reduction in advising time faculty would spend advising students on program and course selections as the system could learn more about the student than the student might share with a faculty member.

The next steps are to examine more algorithms. While k-means is fast, the current prototype of student data has a limited database size of only 100,000 fabricated users. And, the results for actual matches were disappointing after seven interests. As stated before there are many types of underlying algorithms used for recommendation systems. Thus several additional algorithms including hybrid models need to be tested to find the most accurate system. Finally the system needs to be tested on live data.

Thought also needs to include possible schedules because one of the major problems with program planning is courses are often scheduled at the same time which also causes problems when students pre-select a list of classes. The overlap of class times often increases student dissatisfaction.

Another major factor in the system is the probability to run live with the live data. The concern for the administration is student privacy and degradation of performance of the current student information management system. The system at Minnesota State University, Mankato is an Oracle system and contains not just information on Minnesota State Mankato students but students in the entire Minnesota State Universities and College System. The proposed system uses student evaluations of faculty as well as student GPA and majors as part of the algorithm. Although the data used for the algorithm and the itemset cannot be traced back to an actual student, there might be concern about the ultimate possibility. For that reason, discussions are on-going with the university administration and the ITS staff about how to implement such a system with

considerations of data privacy as well as meet the requirements of an on-going live systems.

REFERENCES

1. Alba, J., Lynch, J., Weitz, B., Janiszewski, C., Lutz, R., Sawyer, A., et al. (1997). Interactive home shopping: Consumer, retailer, and manufacturer incentives to participate in electronic marketplaces. *Journal of Marketing, 61*, 38–53.
2. Jacoby, J., Speller, D. E., & Berning, C. K. (1974). Brand choice behavior as a function of information load: Replication and extension. *Journal of Consumer Research, 1*, 33–42.
3. Diehl, K. R., Kornish, L. J., & Lynch, J. G., Jr. (2003). Smart agents: When lower search costs for quality information increase price sensitivity. *Journal of Consumer Research, 30*, 56–71.
4. Ansari, A., Essegaier, S., & Kohli, R. (2000). Internet recommender systems. *Journal of Marketing Research*, 363–375.
5. Iacobucci, D., Arabie, P., & Bodapati, A. (2000). Recommendation agents on the Internet. *Journal of Interactive Marketing, 14*(3), 2–11.
6. Schafer, J. B., Konstan, J., & Riedl, J. (1999, November). Recommender systems in e-commerce. *Proceedings of the ACM Conference on Electronic Commerce*, 158–166.
7. Shardanand, U.,&Maes, P. (1995). Social information filtering: Algorithms for automating "word of mouth." In *Proceedings of the Conference on Human Factors in Computing Systems*, I. Katz, R. Mack, L. Marks, M.B. Rosson, and J. Nielsen, Eds., CHI '95, (pp. 210–217). New York: ACM Press.
8. West, P. M., Ariely, D., Bellman, S., Bradlow, E., Huber, J., Johnson, E., et al. (1999). Agents to the rescue? *Marketing Letters, 10*, 285–300.
9. Häubl, G., & Trifts, V. (2000). Consumer decision making in online shopping environments: The effects of interactive decision aids. *Marketing Science, 19*(1), 4–21.
10. Belkin, N. J. and Croft, W. B.: 1992, 'Information Filtering and Information Retrieval: Two Sides of the Same Coin?' *Communications of the ACM* **35**(12), 29-38.
11. Lang, K.: 1995, 'Newsweeder: Learning to filter news'. In: *Proceedings of the 12th International Conference on Machine Learning*, Lake Tahoe, CA, pp. 331-339.
12. Burke, R. Hybrid Recommender Systems: Survey and Experiments. *User Modeling and User-Adapted Interaction*. 12(4), pages 331-370.
13. Resnick, P., Iacovou, N., Sushak, M., Bergstrom, P., and Riedl, J. ''GroupLens: An Open Architecture for Collaborative Filtering of Netnews'', Proceedings of the CSCW 1994 conference, Chapel Hill, N C October 1994.
14. Towle, B. and Quinn, C.: 2000, 'Knowledge Based Recommender Systems Using Explicit User Models'. In *Knowledge-Based Electronic Markets, Papers from the AAAI Workshop*, AAAI Technical Report WS-00-04. pp. 74-77. Menlo Park, CA: AAAI Press.
15. Booker, Q. (2009) Automating Word of Mouth to Recommend Classes to Students: An Application of Social Information Filtering. *Journal of College Teaching and Learning, 6 (3)*.
16. Booker, Q., Kitchens, F. L. , & Rebman, C. (2007). A Student Course Recommender System Using Linear Classifiers: An Experimental System. *Western Decision Sciences Institute Annual Conference*.