New Classification Algorithms for Developing Online Program Recommendation Systems

Thomas Meller
Centre for Educational & Information Technology
Douglas College
New Westminster, Canada
mellert@douglas.bc.ca

Eric Wang
School of Business, Athabasca University
Athabasca, Canada
ericw@athabascau.ca

Fuhua Lin
School of Computing and Information Systems,
Athabasca University,
Athabasca, Canada
oscarl@athabascau.ca

Chunsheng Yang
Institute for Information Technology,
National Research Council of Canada
Ottawa, Canada
Chunsheng.yang@nrc.ca

Abstract -- This paper presents two novel nearest-neighbor-like classification algorithms for program recommendation in a Web-based system, which provides a program planning service to academic advisors and students of post-secondary institutions. To evaluate the accuracy of classification for program recommendations generated by our algorithm, a statistical study was conducted through comparing our algorithm against two well-known classification algorithms, the Naïve Bayes algorithm and the J48 algorithm, for making recommendations to students based on their academic history. The study shows that our proposed nearest-neighbor-like algorithms outperform the two well-known classification algorithms in terms of student classification success rate when there is uncertainty present in the data.

Keywords: classification algorithms, program planning, recommendation systems, data mining

I. INTRODUCTION

Program planning is an important educational academic advising service to students for a post-secondary institution that offers programs of study that lead to credentials. In order to earn a credential, a student must complete the requirements of a corresponding program, which can be modeled as a matrix of courses and a set of program regulations. For a student, an important question regarding program planning is,

"What program does the school offer?"

This question is often combined with that of,

"*In the myriad of program offerings, what program is the most suited for me*."

There are several aspects to program planning. First, program planning involves investigating what the interests of a student are and providing advising service to students so that their educational needs can be met with appropriate programs. For example, if a student has appropriate academic history but has not decided which program to pursue, then program planning involves assisting the student to determine what program is most suitable to her need and academic background and sequentially to map out which courses can be used towards the student's program of study. To this end, it requires first calculating the requirements that are needed to complete a program. For some programs, granting course substitution is also needed.

Second, program planning often requires a graduation audit that ensures that a graduating student has completed all the requirements of the student's program are met and the student is ready to receive his/her credential. For the most part, graduation audit involves ensuring that a minimum grade requirement for each course was met, a minimum grade point average was achieved, and the total number of credits was satisfied.

Third, a school is always interested in routing students into programs because it is important for the school to track the interest of students in program studies so as to ensure that adequate classroom seats are available, appropriate instructors are available, and an appropriate schedule is produced for the students.

As it is very labor intensive to personally help each student to work out an individualized program of study, an interactive online program planning system is needed to classify students into programs based on their interests and academic history.

With the above necessary functionary components, a program planning system has to have data mining capacity in order to function. This paper develops novel nearest-neighbor-like classification algorithms for a Web-based program recommendation system that provides program planning services to academic advisors and students of post-secondary institutions.

The main contribution of the paper is two new classification algorithms developed for program recommendation based on the nearest neighbor algorithm. The new algorithm can classify students into programs more successfully. We conducted a statistical study that compares

978-0-7695-3528-9/09 $25.00 © 2009 IEEE
DOI 10.1109/eLmL.2009.19

IEEE computer society

the accuracy of the nearest-neighbor-like algorithm against two well-known classification algorithms.

The remainder of the paper is organized as follows. The literature is reviewed in Section II. Section III presents the proposed algorithms for classification. Section IV explains the model for statistical study. Section V discussed the statistical results. Section VI is the discussion and Section VII concludes the paper with future work.

II. LITERATURE REVIEW

When matching a student's academic history against programs, one possible solution is to use the data mining technique classification. One method of classification is using conditional rules and this method is known as rule-based classifiers. An example of a rule-based classifier is the J48 algorithm which is an implementation of the C4.5 revision 8 classification algorithm [5]. Another branch of classification techniques uses inferences as its basis of learning and evaluation. Two examples of the later branch of classification are the Naïve Bayes algorithm and the Bayesian Network algorithm; both algorithms use conditional probabilities. However, the Naïve Bayes algorithm can only handle simple distributions while the Bayesian Network algorithm provides more sophisticated method of inference [5]. They are both eager learners.

One of the earliest classification algorithms is the Nearest Neighbor algorithm that uses a similarity measure to calculate which classes are the closest to the object in question [2]. Typically, a Euclidean or Manhattan similarity measure is used for calculating the similarities between objects although any sensible similarity method can be used. Under the similarity method, each attribute is also assigned an equal weight [4]. Kumar et al. (2006) state that "The justification for using the nearest neighbor algorithm is best exemplified by the following saying, 'If it walks like a duck, quacks like a duck, and looks like a duck, then it's probably a duck.'" [2].

To be effective, distance measures, and similarity measures in general, do not have to satisfy all the properties for a measure [3]. However, measure properties do define how a distance measure behaves. If a measure is well-defined, then it can be used to compare an object to classes within a database with some measure confidence.

A common measure that is used to compare two text objects x and y is the Cosine Similarity measure, which measures the angle between x and y. Kumar et al. mention that this similarity measure, when normalized, may not be adequate if the magnitude between two objects also is an important consideration.

The Cosine Similarity measure uses a vector space model. In the vector space model, documents are represented as a vector of words [4]. When a query is performed against a collection of documents, the query phrase itself is represented as a vector as well [4]. A similarity measure is computed and used to compare the query vector and the document vector. The documents are ranked accordingly.

III. THE PROPOSED ALGORITHMS

A. Similarity Algorithms

First, we propose two similarity algorithms for classification for program recommendation: the Program Similarity Algorithm 1 and the Program Similarity Algorithm 2. These two similarity algorithms used the Curriculum, Advising, and Program Planning (CAPP) module in Banner, a commercial Student Information System, to derive their rules. CAPP is a repository for program definitions and rules which are the basis for academic program functions such as graduation audits. Embedded under these two similarity algorithms, we developed a nearest-neighbor-like classification algorithm to classify students against a program. The nearest-neighbor-like classification algorithm uses a simple calculation to derive a similarity measure and then uses the calculated similarity measure value to rank program matches in descending order.

We assume that a program consists of a finite set of study areas $\{A_1, A_2, \ldots, A_m\}$. An area consists of a finite number of required courses. Elective courses can be defined as well if a program requires so.

- Program Similarity Algorithm I uses both *required courses* and *elective courses* in its calculation;
- Program Similarity Algorithm II uses only *required course* in its calculation.

The formula for calculating a partial similarity measure within an area is as follows:

$$CMA = a / c \tag{1}$$

Where, CMA = credits met in an area of study,
a = number of credits a student has met within an area of study,
c = total credits number of required within an area of study.

To calculate the similarity measure value between a program and a student's academic history, all the areas within a program are considered with the following equation:

$$SPS = \sum_{i}^{m} \left(\frac{a_i}{c_i} x \frac{c_i}{b} \right)$$

Which, simplifies to,

$$SPS = \sum_{i}^{m} \frac{a_i}{b} \tag{2}$$

Where, SPS = similarity between a program and a student

c_i = credits met in area A_i,
a_i = total credits defined within the area A_i,
b = total credits needed within a program,
m = total number of areas within a program.

In this research all the programs were compared when initiating a classification process for a student. It should be possible to develop a more robust and deeper concept hierarchy. Abstract concepts would be represented in the top level nodes of a tree, while the more concrete concepts are represented in the leaves of a tree. A search through a tree would stop at a more abstract level of a tree if the node does not meet a minimal value in the similarity measure. This would prevent the algorithm to compare everything in a tree; thereby, improving performance.

B. Considerations

There were three considerations for developing this new algorithm.

First, it is not necessary for the new algorithm to have a learning phase in this problem domain because the program rules and regulations have already been defined. Moreover, if one is developing rules from academic history, one should expect noise within that data making it harder to derive error free rules. Academic history not only records the rules of a program but also the evolution of program changes, keying errors, and special circumstances – it is not uncommon to grant students exemptions for program requirements based on an individual rationale.

Second, it is desirable to present multiple program suggestions to a student based on her academic history and interests. Although it is possible to do this with existing classification algorithms, suggesting multiple programs will be easier to implement with this new algorithm. As an example, if one were to use a J48 algorithm for suggestions, after each suggestion, retraining the decision tree with previous program suggestions excluded from the training data is necessary in order to calculate new suggestions.

Third, a classification algorithm used in this problem domain needs to perform even if there is noise within the input data. Students often take courses that are unrelated but, of interest to them, before they commit to a program of study.

C. Properties of the Proposed Program Similarity Measures

Equation (2) is a ratio-based similarity measure and satisfies all the properties of a measure. First, it is well-defined because the distance a student has to a program is always a positive real number between 0 and 1. Second, it is commutative because it does not matter if a program is compared to a student or vice versa; although it is more natural for students to be compared against program definitions. Third, under most circumstances, it does support the Triangle Inequality Property because if a student is working on Program A, and then that student decides to work on Program B, that student may lose some credits that have already been earned. The Triangle Inequality Property may not hold when a student transfers from a program that required a larger number of credits, or to a program that does not have a specified field of study such that all college level course credits are recognized and counted into that program. The distance between the student and program B is thus usually larger when traveling through program A. The measure has an Identity Property of 1. If the student has met all of the requirements for a program, then the calculation turns out to be 1 rather than 0; hence, the measure can be categorized as a similarity measure.

D. PROGRAM RULES

As was mentioned before, the Program Similarity One and Program Similarity Two algorithms depend on the rules that were defined within the Banner Student Information System. At BCIT, the CAPP module houses the official program definition and rules for the institution. The program definitions are periodically audited by the various schools of study to ensure accuracy. This audit also involves an approval process with high level management. An example of a program matrix is shown in Table I.

Table I: Advanced Java Development Program Matrix (Associate Certificate)

Required Courses		**Credits**
COMP 2611	Intermediate Java	3.0
COMP3621	Advanced Java	3.0
COMP3631	Java Web Applications	3.0
COMP3711	Object Oriented Analysis and Design	4.0
COMP4620	Java Databases	1.5
COMP4631	Advanced Java Web Applications	1.5
COMP4652	Enterprise Application Development with JEE	3.0
Elective Courses		
COMP1911	JavaScript Workshop	1.0
COMP2011	AJAX Workshop	1.0
COMP2612	Java - Rapid Application Development 1	1.5
COMP3619	Java User Interfaces	1.5
COMP4653	Enterprise Application Architecture and Design with J2EE	3.0
COMP4690	Java Project	3.0

IV. STATISTICAL STUDY

During the execution of this project, there were two methods for generating rules for classification. One method used a Java software package called WEKA to train the J48 and Naïve Bayes algorithm [5]. In this method, the training data comprised of the academic history of previously graduated students. The rules were derived from operational data. The Program Similarity One and Program Similarity Two algorithms did not require training. They derived the rules of a program from domain experts – program

definitions provided by the Banner Student Information System. All four algorithms had to predict the graduation outcome of a set of graduated students.

A statistical study was performed to evaluate the Program Similarity One and Program Similarity Two algorithms against the J48 and Naïve Bayes algorithms. Students from one graduation set, or term, were used to compare against the various methods. In addition, programs from three separate schools were used: the School of Computing and Academic Studies, the School of Health Sciences, and the School of Business of BCIT (British Columbia Institute of Technology) of Canada. The Program Similarity One and Program Similarity Two algorithms were hand coded. WEKA [5], an Open Source data mining package written in Java, was used to generate statistics for the J48 and Naïve Bayes algorithms. Cross validation tests were used when generating statistics for the J48 and the Naïve Bayes algorithms.

Hypothesis testing was conducted during this project. The four algorithms mentioned were tested against three different populations – the three different schools. In all cases, the same set of students was tested against each algorithm. The equation that was used to calculate a z-score was as follows:

$$Z = \frac{P1 - P2}{\sigma_{P1-P2}} \quad (3)$$

Where, Z = z-score,
P1 = sample 1 proportion,
P2 = sample 2 proportion,
σ_{P1-P2} = standard deviation of the two samples.

The standard deviation was calculated as follows:

$$\sigma_{P1-P2} = \sqrt{pq\left(\frac{1}{N_1} + \frac{1}{N_2}\right)} \quad (4)$$

Where, $p = \frac{N1P1 + N2P2}{N1 + N2}$
N1 = sample 1 size
N2 = sample 2 size
p = the sample proportions
q = 1-p

V. STATISTICAL RESULTS

When conducting the statistical study to evaluate the various algorithms, a large sample size was used. The sample size for the School of Business of BCIT was 1066 students. The sample size for the School of Computing and Academic Studies of BCIT was 124 students, while the sample size for the School of Health Sciences of BCIT was 234 students. Each algorithm was evaluated against the same sample of students.

The sample consisted of all the students from a single graduating year from the respective schools. The three schools of study used in the sample were randomly chosen.

A. Accuracy Rates

Overall, the proposed Program Similarity Algorithm One and the Program Similarity Algorithm Two outperformed the J48 and Naïve Bayes algorithms in which the most pronounced difference was within the School of Computing and Academic Studies. The Program Similarity Algorithm One and Program Similarity Algorithm Two had an accuracy rate of 79.83% and 89.29% respectively within this school.

Table 2: Percentage of Students Properly Classified

School	Program Similarity 1	Program Similarity 2	Naïve Bayes	J48	Sample Size
Business	83.11%	81.89%	74.86 %	74.4 8%	1066
Computing & Academic Studies	79.83%	86.29%	37.10 %	32.2 6%	124
Health Sciences	95.73%	91.45%	88.89 %	90.1 7%	234

The Naïve Bayes algorithm had an accuracy rate of 37% while the J48 algorithm had an accuracy rate of 32.26%. The easiest school to classify was the School of Health Sciences. The Program Similarity One and Program Similarity Two algorithms performed at a 95.73% and 91.45% accuracy rate respectively. The Naïve Bayes algorithm had an accuracy rate of 88.89% while the J48 algorithm had an accuracy rate of 90.17%. The School of Business had accuracy rates that were between the two schools mentioned above. More details are provided in Appendix A.

The results from the School of Computing and Academic Studies are explainable. The data has a lot noise in it due to internal and external equivalencies that are not recorded within the database. Course requirements can easily be submitted for similar courses with the permission of the institution. As an example, a requirement for an *Introduction to Programming in Java* course can sometimes be substituted for an *Introduction to Programming using Visual Basic* course. Due to these course substitutions, regular classification algorithms do not perform well. The rule based nearest-neighbor-like algorithms, Program Similarity One and Program Similarity Two algorithms perform better because they calculate the closest fit based on predefined rules from domain experts.

Although the situation at the School of Health Sciences is contrast to that at the School of Computing and Academic Studies, its results are explainable as well. There is little noise within this data set stemming from course substitutions because they are not as prevalent within this school. For instance, a requirement such as an Introduction to Microbiology cannot be substituted for a course in Anatomy

and Physiology for Cardiology. Regular classification algorithms perform better because there is less uncertainty within the data.

Under certain circumstances, the J48 and the Naïve Bayes algorithm outperformed the Program Similarity One and Program Similarity Two algorithms. The most obvious case is the Bachelor of Business Administration credential. The Program Similarity algorithms could not classify these students while the Naïve Bayes and J48 algorithms had a classification rate of 100%. The rationale for this is that the rules for the Similarity One and Similarity Two algorithms were poorly defined in the program definitions while the J48 and Naïve Bayes algorithms were trained from academic history.

Hypothesis testing was conducted within this project and sometimes the null hypothesis held true. The lowest z-score was between Program Similarity Two algorithm and the J48 algorithm within the School of Health Sciences. The z-score was .48 which leaves roughly an 18.4% chance that the Program Similarity Two algorithm produces better results. Therefore, in this case, the null hypothesis holds true and there probably isn't a difference in accuracy. Again, the School of Health Sciences had the second lowest z-score when comparing the performance between the Naïve Bayes algorithm and the Program Similarity Two algorithm. It had a z-score of .93 which leaves roughly a 32.28% chance that the algorithms are different in classification accuracy.

For some samples, the alternate hypothesis held true. The highest z-score was between the Program Similarity Two versus the J48 algorithm. It had a z-score of 8.659. This z-score occurred within the School of Computing and Academic Studies. With such a high z-score it can be said that there is a 99.99% chance the Program Similarity Two algorithm outperforms the J48 algorithm.

The statistical results have an interpretation. The Naïve Bayes and J48 algorithms produce accurate rules when there is no uncertainty within the data. These algorithms perform well in the School of Health Sciences because course substitutions are not prevalent. However, when there is uncertainty within the data, the Program Similarity algorithms perform better. These algorithms allow for noise.

Domain experts validated the statistical results. They agreed that the School of Health Sciences does not use a lot course substitution while the School of Computing and Academic Studies does. They also agree that the School of Business is somewhere between these two schools for a measure of uncertainty. The statistical data does back up what domain experts know.

B. Classification Performance

The Program Similarity One and Program Similarity Two algorithms performed quite well. On average it takes 110 milliseconds to classify a student's academic history against all 88 programs that the School of Business has to offer. Said in another way, it takes on average 13.5 milliseconds to classify a student's academic history against one single program. The computer that was used in these experiments was an Intel Dual processor running at a clock speed of 2.79 GHz with 1 Gigabyte of memory. The Program Similarity method mentioned above is a $O(N)$ algorithm. Its complexity is linear.

VI. DISCUSSION

We developed a prototype of Web-based program recommendation system to test the effectiveness of the approach using the Term Frequency and Inverse Document Frequency formula to create a text index for keyword-based program search and proposed and developed a novel nearest-neighbor-like algorithm for program recommendation [7]. It is our opinion that the prototype, if it were put into production and integrated with computer-aided advising system such as e-advisor developed at Athabasca University of Canada [8], would add value to the student. The prototype empowers the student to investigate programs of interest for themselves. An academic advisor is at times restricted by the knowledge and questions a student is asking. If a student does research beforehand, the program advisor may be able to help the student with more in-depth questions. The prototype answers questions about interests through key words. It compares programs, shows the courses attached to programs, it answers basic matriculation questions, and it gives suggestions to programs they may want to investigate based on their academic history. Theoretically, the idea of the prototype may alleviate frustration in getting answers; thereby, allowing for a healthier relationship between the student and the institution.

The statistical study compared four classification algorithms and showed that two algorithms we proposed that were similar to a nearest neighbor algorithm but, used rules from domain experts, outperformed the two well-known algorithms that derived the rules from operational data within a very specific problem domain. The J48 algorithm and Naïve Bayes algorithm had higher rates of error when there was uncertainty within the data. In order for users to accept suggestions, they need to make sense; thus, the accuracy rate is important. Rules provided by domain experts outperform computer derived rules in the problem area of program classification and should increase the confidence rate a user has from computer generated suggestions.

When a Web product helps a customer with their goals and tasks, a relationship is built with the customer. This may seem odd because there is no human to human interaction under this relationship. It is human to go back to a Web site that offers good personalization. Making informed decisions about an education direction eases worries that a student may have. After all, pursuing a program of study is a huge investment in resources.

As was outlined above, the end result of some text indexing methods is a similarity measure between the input query and the indexed documents. It ranks the highest fit of an input query to a collection of documents. This is very similar to a nearest neighbor algorithm. Is it possible to

create a general purpose engine that can classify both text and students into programs?

Such a general purpose engine would still depend on the accuracy of its rules. It is clear from this research project that when rules are well defined within a problem domain, classification accuracy improves. In this project, the well defined rules were built by people. Is it possible for domain experts to facilitate a classification process? Perhaps accuracy would improve if domain experts provide exemplar rules about a problem domain and machine learning techniques are used to build rules around those exemplar rules. In other words, using machine learning techniques would fill in the details. Using this problem domain, domain experts would provide courses that are unique and required for each program, the computer could do data mining and provide the remaining courses that are needed. The computer could also provide rules for equivalent courses.

VII. CONCLUSION AND FUTURE WORK

We have showed that it is possible to develop a better self-help product for exploring programs at a post-secondary institution. Classification was used to identify what programs a student may fit into. A search engine was constructed to allow a student to search on areas of interest.

It is our opinion that machine-learning algorithms are very important in personalizing Web products. A common technique of indexing documents is a form of classification. The input query is classified against a collection of documents. In this project's problem domain, a graduation audit is again a form of classification. If a student has met all the requirements of a program, there is an exact match against a program. Understanding what class an object belongs to is a basic function that a computer system should have. It would lead to useful functionality helping users with their goals and tasks.

Specific to this problem domain, more research needs to be done with the classification algorithm. The performance measures were adequate when using just the School of Businesses programs. However, to involve all of the schools would require undesirable delays in a Web application. One possible solution is to classify a student into a small set of schools before the program classification process takes place. The concept of a school is more abstract than a program. Lastly, we hope to conduct a statistical study on the effect of using rules within an information retrieval process and compare the difference of text indexes in increasing the accuracy of a query with and without using rules in its processing.

REFERENCES

[1] Brendan, F., & Duek, D. (2007).*Affinity Propagation: Clustering by Passing Messages Between Data Points.* Retrieved October 11, 2007 from http://www.psi.toronto.edu/affinitypropagation/

[2] Kumar, V., Steinbach, M. & Tan, P. (2006*).Introduction to Data Mining.* Montreal: Pearson Addison Wesley.

[3] Berry, M., & Linoff, G. (2004).*Data Mining Techniques: For Marketing, Sales, and Customer Relationship Management.* USA: John Wiley & Sons. Retrieved February 20, 2008 from http://0-site.ebrary.com.aupac.lib.athabascau.ca/lib/athabasca

[4] Kamber M., Han, J. (2006). *Data Mining: Concepts and Techniques, Second Edition.* San Francisco: Morgan Kaufmann Publishers.

[5] Frank. E., & Witten, I. (2005). *Data Mining: Practical Machine Learning Tools and Techniques, Second Edition.* San Francisco: Morgan Kaufmann.

[6] Jones, S. K., (1972). *A statistical interpretation of term specificity and its application in retrieval,* Journal of Documentation, 28 (1), 11-21. http://www.soi.city.ac.uk/~ser/idfpapers/ksj_orig.pdf

[7] Meller, T., (2007), Constructing a Prototype of an Intelligent Web Application for Program Planning, MSc IS Thesis, Athabasca University.

[8] Lin, F., Kinshuk, R. McGreal, S. Leung, D. Wen, F. Zhang, Q. Li, X. Liang, (2008), e-Advisor: A Web-based Intelligent System for Academic Advising, *International Transactions on Systems Science and Applications, (ITSSA),* 4(1), March 2008, 89-98