Using Student Performance Predictions in a Computer Science Curriculum

A.T. Chamillard
Computer Science Department
University of Colorado at Colorado Springs
Colorado Springs, CO 80933-7150
719-262-3150
chamillard@cs.uccs.edu

ABSTRACT
Professors often develop anecdotal guidelines about how each student's past performance in their academic major relates to their performance in later courses. While these guidelines can be useful, a more formal statistical analysis of these relationships can provide valuable insight into predicted student performance, which can help professors guide their students to focus on potential areas of difficulty. In addition, such analyses can identify which courses are key indicators of later performance in the major. This additional insight into the relationships between the courses in the curriculum can help professors implement curriculum changes and measure the effects of those changes. In this paper, we present the results of such an analysis for computer science majors at the U.S. Air Force Academy.

Categories and Subject Descriptors
K.3.2 [**Computers and Education**]: Computer and Information Science Education – *computer science education, curriculum.*

General Terms
Measurement, Performance, Experimentation.

Keywords
Student performance, performance prediction, predictive modeling, computer science curriculum.

1. INTRODUCTION
Predicting student performance in a particular course, or even on assessments within a course, is a difficult but useful undertaking. Given such predictions, a professor can help focus student effort on potential problem areas for particular students given their performance in previous courses. In addition, curriculum committees can use prediction results to guide changes to the curriculum and evaluation of the effects of those changes.

Typically, student performance predictions and identification of the key courses in an academic major are based on anecdotal evidence (if they are developed at all). While anecdotal evidence is certainly useful for some purposes, it is also reasonable to use more formal statistical analysis techniques to develop models to help predict student performance and to identify key courses.

Prior efforts to predict student performance in various majors and institutions are documented in the literature. Thomas examined performance prediction in an introductory physics course [11], indicating that prior performance is more effective for predicting student performance in that course than a diagnostic test. Felder et al [7] predicted performance in an introductory chemical engineering course using a variety of factors, including performance in previous courses. Other researchers have built and validated models predicting introductory computing aptitude [8] and used high school performance data and other demographic factors to predict introductory programming performance [2, 3, 10] and retention in the computer science major [4]. The results presented here differ from previous performance prediction efforts because they include formal predictive models for each required class in the computer science curriculum and provide insights that span the entire computer science curriculum rather than focusing on a selected course or class year.

In addition to offering some predictive capabilities, the results of a formal statistical analysis can also provide educators with insight into the relationships between the courses in the curriculum. This insight can facilitate curriculum modification efforts in a number of ways. For example, given limited resources to accomplish curriculum changes, educators can focus their efforts on those courses that have or should have the strongest effect on student performance in later courses. As curriculum changes are implemented, changes in the statistical relationships between student performance in different courses can be monitored to ensure the desired effects are achieved.

In this paper, we present the results of a formal statistical analysis using data for 285 computer science majors spanning 7 class years at the U.S. Air Force Academy, called USAFA hereafter. While some of the specific insights resulting from this analysis may only be applicable to the computer science major at USAFA, the analysis approach, use of the results, and implementation issues are more generally applicable to both other academic disciplines and other educational institutions.

The following sections briefly discuss the required courses included in the computer science major at USAFA, present our analysis approach, provide the results of our statistical analysis, discuss how the results can be used in practice, and present our conclusions.

ITiCSE'06, June 26–28, 2006, Bologna, Italy.

2. REQUIRED COURSES

In this section we present the required major's courses taken by computer science majors at USAFA. Students also take optional courses to complete the major, but we only consider required computer science courses in our analysis here. The computer science program at USAFA was CSAB-accredited during the period under analysis (it is now accredited by ABET), and topic coverage closely follows ACM curriculum guidelines [1].

We note that the content and sequence of major's courses at USAFA is continually examined and modified as necessary. The courses listed here were taken by the students included in the dataset, but the course contents, the semester, and even the year in which they are taken are subject to change as the major evolves. A summary of the required courses is provided in Table 1.

In general, all students in the dataset were required to take all of the above courses. There is one exception to this rule, however: CS 463, the Introduction to Database Management course, which was only a required course for 4 out of the 7 class years included in the dataset.

3. ANALYSIS APPROACH

In this section we describe the statistical analysis techniques we used to analyze performance data and to generate our predictive models. We present our technique for using linear regression to generate the predictive models, discuss our analysis of the resulting models and their associated parameters, and describe the correlations we generated for further analysis.

For the purposes of this paper, course assessments are defined as graded activities in the course, such as programming assignments and exams. Student performance on the course assessments is defined as the student's overall percentage on those assessments. Overall student performance in a particular course is captured by both the student's overall course percentage and the letter grade the student earned in the course.

The primary goal of our statistical analysis was to develop predictive models for each of the course assessments and the overall student performance in each course using student performance in prior courses as predictor variables. We can then use those predictive models to predict student performance in a particular course so we can guide students to focus their studies on specific potential problem areas. Additional goals included gaining further insight into the relationships between computer science courses and providing a mechanism for monitoring the impact of curriculum changes.

3.1 Building the Models

Linear regression models can be used as approximations of the functional relationship between a predicted value and a set of predictor variables [9]. We used linear regression to build our predictive models for each assessment and overall performance in each course.

In linear regression, the predictive model is of the form $y = \beta_0 + \beta_1 x_1 + \ldots + \beta_i x_i$, where y is the predicted value, each x_i is a predictor variable, and each β_i is a coefficient calculated using linear regression. The regression coefficients are calculated using a linear least squares fit to the data. We note that, in general, larger coefficients (positive or negative) indicate a stronger predictive effect from the associated predictor variable.

3.2 Analyzing the Models

To determine how well a predictive model fits the data used, we need some measure of how well the model captures the variance in the data. For linear regression, the standard measure of fit is the Multiple Correlation Coefficient Squared, R^2, which measures how much of the variance in the predicted value is captured by the predictive model. R^2 ranges from 0.0 to 1.0, with a magnitude near 1.0 indicating that the model explains most of the variance in the data, which in turn implies that the model provides good predictive power.

The linear regression technique we used assumes that the errors (e.g., residuals) in the model are independent, have zero mean, constant variance, and follow a normal distribution [6]. These assumptions can be checked using plots of the standardized residuals against the predicted response variables.

The structure we would expect to find in these plots, given our assumptions about the errors in the model, is essentially a horizontal line with residual values scattered randomly above and below zero. If we find that the "spread" of the residuals increases or decreases as the value of the response or predictor variable increases, we should suspect that the variance is not constant. While more formal statistical techniques have been proposed for

Table 1. Required Computer Science Courses

Course	Course Title	Semester	Year
CS 225	Fundamentals of Computer Science (CS1)	Fall	Sophomore
CS 359	Programming Paradigms	Spring	Sophomore
CS 326	Foundations of Computer Science	Fall	Junior
CS 351	Computer Organization and Architecture I	Fall	Junior
CS 356	Computer Organization and Architecture II	Spring	Junior
CS 380	Algorithms and Data Structures	Spring	Junior
CS 453	Software Engineering I	Fall	Senior
CS 463	Introduction to Database Management	Fall	Senior
CS 483	Operating Systems	Fall	Senior
CS 454	Software Engineering II	Spring	Senior
CS 467	Computer Networks	Spring	Senior

checking these assumptions, visual inspection of the residual plots is generally sufficient for recognizing serious violations of the assumptions [6].

3.3 Generating Correlations

Although the predictive models can yield valuable insight into the relationships between courses in the major, it may also be enlightening to measure these relationships directly. We can use the Pearson correlation coefficient (r) to examine the linear relationship between two variables. The coefficient ranges from –1.0 to 1.0, with a coefficient magnitude close to 1.0 indicating a strong linear relationship and a magnitude close to 0.0 indicating a weak linear relationship.

To gain additional insight into the relationships between the courses in the major, we calculated correlation coefficients for pairings of the course grades for each course. Analysis of those correlation coefficients is provided in the following section.

4. ANALYSIS RESULTS

4.1 Dataset Description

The dataset used for the statistical analysis presented in this paper consists of 285 computer science majors from the Class of 1995 through the Class of 2001. Of these 285 students, 175 students graduated with a computer science degree; the other 110 students either changed to a different major or left USAFA without graduating. When building the predictive models for a particular course, we included all students who took that course, whether or not they ultimately graduated with a computer science degree.

The dataset contains 117 measurements, including percentages for the assessments in each course, the overall percentage in each course, and the course letter grade in each course. We encoded the course grade using standard GPA (Grade Point Average) values for letter grades (A = 4.0, A- = 3.7, B+ = 3.3, etc.).

4.2 Predicting Performance

For the predictive models we built, we only considered predictive models with an R^2 greater than or equal to 0.500 as sufficiently powerful to be potentially useful and, therefore, to merit further discussion. None of the residual plots for any of the models discussed below contained patterns indicating violation of the linear regression assumptions.

We did not develop any models for our CS1 course because there are no previous courses to provide performance data. For the remaining 10 courses, we built 99 models, using the percentages for the course assessments (assignments, exams, etc.), overall course percentages, and course grades as the predicted values. For the predictor variables, we used only course grades from courses completed prior to the course for which we were building predictive models. Our initial goal was to use all assessments for prior courses in the models as well, but incomplete data precluded that approach. Our linear regression approach requires that the data for a particular student contain all predictor variables from previous courses for that student to be included in the modeling, which would lead to significant data losses in the modeling. In fact, we only had complete data for one student (out of 285) for all the courses being analyzed. We were able to gather course grade data from transcripts provided by the registrar, but those transcripts do not contain detailed course data.

To summarize, when predicting performance in each course we used previous course grades as the predictor variables. We did, however, attempt to predict performance on all assessments in that course despite the fact that we typically did not have complete course assessment data for all students in the dataset.

There were a number of courses for which none of the models built for the assessments, overall percentage, or course grade had reasonable predictive power. Those courses included Programming Paradigms (CS 359), Computer Organization and Architecture II (CS 356), Introduction to Database Management (CS 463), and Software Engineering I (CS 453) and II (CS 454).

We were able to build one or more predictive models with reasonable predictive power for the following courses: Foundations of Computer Science (CS 326), Computer Organization and Architecture I (CS 351), Algorithms and Data Structures (CS 380), Operating Systems (CS 483), and Computer Networks (CS 467). While space limitations preclude presentation of the actual models, we note that we developed 7 models with reasonable predictive power. Six of those models predicted course percentage or course grade, while the seventh model predicted performance on the Operating Systems programming assignments.

4.3 Correlation Results

While the predictive models discussed above provide some insight into the relationships between the courses in the major, directly calculating correlation coefficients between the course grades can also be enlightening. The resulting coefficients ranged from 0.29 to 0.73, providing interesting insights into several pair-wise course relationships. We also note that if the relatively common anecdotal argument that "stronger students generally do well across the curriculum and weaker students generally do poorly" were true, these coefficients would likely be higher; another argument for using statistical analysis rather than anecdotal observation to evaluate student performance.

It is interesting to note that the strongest three correlations occur between courses that students take in the same semester. Students take CS 351 and CS 326 (r=0.73) in the fall semester of their junior year, CS 356 and CS 380 (r=0.72) in the spring semester of their junior year, and CS 463 and CS 483 (r=0.72) in the fall semester of their senior year. We believe the correct interpretation of this result is that students having a “good” semester do well in most or all of their computer science classes in that semester, while students having a “bad” semester do poorly in most or all of their computer science classes in that semester. We note that, given the structured environment at USAFA, the semesters in which particular courses are taken is a requirement, not a suggestion.

One of the weaker correlations (r=0.47) involves the two courses in the software engineering capstone sequence (CS 453 and CS 454). This could be due, at least in part, to the fact that 50% of a student’s grade in both CS 453 and CS 454 is based on a group project rather than on individual work. This grading policy reduces the effect that a particular student’s past performance would have on that student’s grade in these courses. It is interesting to note that CS 453 and CS 454 are essentially treated as a year long course in terms of material coverage and project

work [5], but there is only a weak correlation between the course grades for these courses. Students are moved into different groups between CS 453 and CS 454, however, so this weak correlation may also be indicative of the significant effect that group performance has on individual course grades in these courses.

5. USING THE RESULTS

5.1 Student Focus

Most of the models with reasonable predictive power can be used to predict a student's overall performance in a particular course given their performance in previous courses. For example, the predictive model for the Overall Percentage in CS 483 (Operating Systems) had the strongest predictive power, with $R^2 = 0.579$ (see Figure 1). A student with a C- (1.7) in CS 380 (Algorithms and Data Structures), CS 356 (Computer Organization and Architecture II), and CS 326 (Foundations of Computer Science) would have a predicted CS 483 percentage of 64%. This is almost certainly a failing grade (D) in the course, so the student could be warned that, based on their past performance, additional effort may be required to successfully complete the course. We note that using only the information that the student earned C- grades in the 3 previous courses, both the student and the instructor might believe that this student could expect a C- from this course as well. The predictive model, however, shows that the student is at higher risk for failure in the course than anecdotal examination of the previous course grade information indicates; although the difference between C- and D is small, it is significant because it is also the difference between passing and failing the course.

This information should not be used as a threat, of course; instead, it should be used to help motivate the student toward expending additional effort to try to succeed in the course. This same technique can be used for the other predictive models discussed above.

Predictive models can be used even more effectively to help guide student efforts when the predicted variable is a course assessment rather than an overall course grade. For example, the predictive model for the CS 483 programming percentage (see Figure 1) indicates that CS 356 (Computer Organization and Architecture II), CS 380 (Algorithms and Data Structures), and CS 359 (Programming Paradigms) grades are significant predictors of performance in this course assessment. A CS 483 student who did poorly in one or more of those classes could therefore be encouraged to expend additional effort on the programming assignments.

Continuing our CS 483 programming percentage example, it can certainly be argued that effective instructors will notice through course assessment mechanisms which students are having trouble with the programming assignments in the course and can offer suggestions to focus the efforts of those students as necessary. The key distinction between that approach and using predictive models is that the predictive models can be used to provide this focus to the students at the beginning of the course, while the latter approach requires that one or more programming assignments be completed before the students who are having trouble can be identified. Using the predictive models lets us provide early focus to those students predicted to have problems in this area rather than waiting until those students have performed poorly on one or more assessments before providing that focus.

It is clear that there is some danger associated with using predictive models to predict student performance in a course. One such danger is that an instructor who tells a student their precise expectations of that student's performance may negatively influence the student's performance, particularly if those expectations are for poor performance in the course or on particular course assessments. Rather than providing a predicted grade to each student at the beginning of the course – such a prediction could even be viewed by students as a "contract" for a particular grade – it would be more prudent for instructors to use the predicted grades to identify the students who may need to expend extra effort on the course or particular assignments in the course. Those students can then be approached for a general discussion about their predicted performance in the course ("Based on your grades in CS 356, CS 380, and CS 359, you'll probably have to spend some extra time on this course") rather than with a specific grade prediction ("Based on your grades in CS 356, CS 380, and CS 359, you'll probably get a D in this course"). While this approach doesn't completely alleviate concerns about students living up (or down) to instructor expectations, it does provide a way to provide early focus to those students who may need it the most.

5.2 Curriculum Changes

The statistical results from this dataset and, more generally, from the overall analysis approach can also be used to help identify potential curriculum changes and to measure the effects of those changes.

The predictive models we generated identified several "key courses" in the major; specifically, those that seem to provide the strongest predictive effect in the predictive models for later courses. In the 7 models discussed above, the CS 359 (Programming Paradigms) course grade appears as a statistically significant predictor variable in 6 of those models: twice as providing the most predictive effect, 3 times as providing the second-strongest effect, and once providing the third-strongest effect. The CS 225 (CS1) course grade appears as a statistically significant predictor variable in 3 of the models: twice as providing the most predictive effect and once as providing the second-strongest effect. All other course grades appeared in three or fewer of the models, with smaller predictive effects. While the fact that CS 359 and CS 225 are sophomore classes indicates that they were included in the modeling efforts for all junior and senior courses, thereby giving us some expectation that they would appear in more predictive models, we note that their predictive power in those models indicates that they are key courses in the computer science major. This is consistent with our intuition about those courses; because they are the first two courses taken by all computer science majors, these courses are

$$CS483\% = 4.900 * CS380Grade + 3.982 * CS356Grade + 2.530 * CS326Grade + 44.684$$

$$CS483Programming\% = 8.395 * CS356Grade + 8.077 * CS380Grade + 7.836 * CS359Grade + 9.431$$

Figure 1. CS 483 (Operating Systems) Predictive Models

structured to cover foundational computer science topics that we expect the students to need and use throughout the major.

Our correlation results also yielded valuable information about the relationships between the courses in the curriculum. For example, we were surprised to find that the correlation between the Computer Organization and Architecture I and II grades was relatively weak. This result is somewhat counterintuitive because these courses represent a two-course sequence on computer organization and architecture. We can use these results as one indication that the two courses may not be as tightly integrated as we originally intended. We can therefore make curricular changes to tie these two courses more tightly together, re-accomplish our statistical analysis after the changes have been implemented, and examine the resulting relationship to gain insight into the effectiveness of our changes (a higher correlation between the grades implies a stronger connection between the courses).

As indicated above, the predictive models and the correlation coefficients are expected to change as curriculum changes are implemented (and as the dataset grows over time as well). Given the dynamic nature of the computer science curriculum, it would be unreasonable to expect a static set of predictive models and other statistical relationships using the approach described above. Instead, we envision an evolution of these relationships as curriculum changes occur and the size of the dataset increases. After the initial analyses are completed, the predictive models and correlation coefficients can easily be updated on an annual basis.

6. CONCLUSIONS

Using student performance data from previous courses and predictive models that predict student performance in a particular course, professors can help focus student effort on potential problem areas in that course. Educators can also use this information to guide their implementation and evaluation of curriculum changes.

In this paper, we reported the use of statistical analysis techniques to build such predictive models. While many of the generated models did not have sufficient predictive power to be useful, the stronger models and other observations from the analysis provide useful insight into the relationships between the various courses.

There are still numerous analyses that can also provide valuable information. While most of the models presented here use only previous course grades as predictor variables, it is possible that more robust models and additional insights could be provided if complete assessment data for each course were also available. To gain these additional benefits, we suggest an ongoing data collection strategy where course assessment data for each course is collected and archived every semester. This approach would ensure that complete data is available for each student, providing the opportunity to include assessment predictor variables in the predictive models rather than limiting the predictor variables to previous course grades. As a separate but related extension, we could include elective courses in our analysis to evaluate the relationships between student performance in required and elective courses and the assessments in those courses.

It would also be interesting to determine whether or not it is possible to predict probable success or failure in the major based on performance in the early courses in the major. These predictions could simply predict "Yes" or "No" for graduation with a computer science degree, or they could be used to predict Major's GPA or final standing among the majors in the same class year. It may also be useful to include other predictor variables (SAT or ACT scores, high school GPA, and so on) as predictor variables in these models. Given predictions based on such models, we could provide students with sound feedback early in the major.

The analysis approach and observations presented here are not limited to either the computer science major or to USAFA; in fact, the author is now implementing this approach at a different university. The approach is sufficiently general that it can be applied to any major at any institution, and other researchers could conduct similar analyses to provide additional observations and recommendations.

7. REFERENCES

[1] ACM/IEEE-CS Joint Curriculum Task Force. *Computing Curricula 1991*. ACM Press, New York, NY, 1991.

[2] Butcher, D.F., and Muth, W.A. Predicting performance in an introductory computer science course. *Communications of the ACM, 28*, 3 (Mar. 1985), 263-268.

[3] Byrne, P., and Lyons, G. The effect of student attributes on success in programming. In *Proceedings of the 6th Annual Conference on Innovation and Technology in Computer Science Education (ITiCSE 2001),* Canterbury, UK, 49-52.

[4] Campbell, P.F., and McCabe, G.P. Predicting the success of freshmen in the computer science major. *Communications of the ACM, 27*, 11 (Nov. 1984), 1108-1113.

[5] Chamillard, A.T., and Braun, K.A. The software engineering capstone: Structure and tradeoffs. In *Proceedings of the 33rd SIGCSE Technical Symposium on Computer Science Education,* Northern Kentucky, KY, February 27-March 3, 2002, 227-231.

[6] Draper, N.R., and Smith, H. *Applied Regression Analysis*. John Wiley & Sons, New York, NY, 1966.

[7] Felder, R.M., Forrest, K.D., Baker-Ward, L., Dietz, E.J., and Mohr, P.H. A longitudinal study of engineering student performance and retention: I. Success and failure in the introductory course. *Journal of Engineering Education, 82,* (1993), 15-21.

[8] Glorfeld, L.W., and Fowler, G.C. Validation of a model for predicting aptitude for introductory computing. In *The Papers of the Thirteenth SIGCSE Technical Symposium on Computer Science Education*, Indianapolis, IN, 1982, 140-143.

[9] Montgomery, D.C., and Peck, E.A. *Introduction to Linear Regression Analysis*. John Wiley & Sons, New York, NY, 1982.

[10] Taylor, H.G., and Mounfield, L.C. The effect of high school computer science, gender, and work on success in college computer science. In *Proceedings of the Twentieth SIGCSE Technical Symposium on Computer Science Education*, Louisville, KY, 1989, 195-198.

[11] Thomas, E.W. Performance prediction and enhancement in an introductory physics course for engineers. *Journal of Engineering Education, 82*, (1993), 152-156.