CS1 student grade prediction: unconscious optimism vs insecurity?

Sónia Rolland Sobral

Abstract— **The difficulties of many students in introductory programming courses and the consequent failure and drop out make it necessary to look for motivation strategies for them to be successful. One of the strategies that is touted in the literature is self-assessment to compromise and motivate students. As we had doubts about the possibility of this strategy, we did an experiment and asked the students to predict the grades of the two tests and the two projects during a semester. Even knowing the correction grid and exercises that involve programming languages, which shows the result to the programmer, we found that the students' forecasts were not very accurate. In the first test we found that the worst students said they were going to get reasonable grades and much better than reality, while the best students thought they had worse grades than they actually had. The other moments of evaluation did not have as severe results, but forecasts continued to be inaccurate. We did tests by gender, by age, for being a freshman or not, for having taken a computer course in high school and for previous knowledge of programming languages: none of these variables proved to be as significant as the students' grades and their corresponding insecurity-fear or optimism-unconscious.**

Index Terms—**CS1; grade predict; introduction to programming; motivation strategies**

INTRODUCTION

CS1 (computer science 1) is the designation widely used for introduction to programming courses in computer science major since ACM's 1978 Computing Curricula [1]. The course belongs to the first semester and is where many students start using programming language to run small programs. Teaching how to program is a task that proves to be complicated since although some students love it and can succeed easily, others feel that it is almost impossible to be able to pass the course, which often leads them to abandon it [2][3]. Failure and dropout rates are traditionally very high [4], which makes any professional responsible and motivated to carry out all tasks in order to help students develop the necessary skills in such a unit [5] [6]. Some methods and strategies have been extensively studied and tried to motivate students who do not have the internal strength to succeed: active methodologies [7] [8] [9], project based learning [10], agile methodology such as SCRUM [11] [12], pair programming [13] and many others [14] [15]. Some researchers suggest the use of self-assessment as a way to rethink their own work and commit students to tasks [16] [17] [18] and to have greater awareness of their own behavior, motivation, and cognition [19]. This article tries to assess the possibility of using self-assessment as one of the strategies in teaching programming to freshman university students. As an exploratory technique, an experiment is made to propose to students that after the exam and after the correction is presented by the teacher, students should do a prediction of the grade they will obtain. The curricular units foresees two tests and two group projects. The grades expected by the students are also indicated by them after making the presentation and discussing the project. There are experiments that use grade prediction at the beginning of the semester [19], during the semester [20], before the exam, others at the end of the exam [22]. The particularity is that this article is associated with a course of introduction to programming in which the results are almost objective and not as subjective as the experiences reported in the literature, usually in psychology or economics. This experience is only experimental and the grades that students say they deserve are not used in the formula for calculating the final classification of the course. This strategy is used very cautiously because the literature indicates that there is a clear tendency for students to be too optimistic and to classify themselves with grades that are not at all what they deserve.

This article is divided into a literature review on grade prediction by students, methodology (course organization, survey methodology and data collection), results, discussion and conclusions.

LITERATURE REVIEW

Self-regulated academic learning emerged in the 80s with the prospect that students become master of their own learning process [22]. Self-regulation is manifested in the active monitoring and regulation of a number of different learning processes, e.g. the setting of, and orientation towards, learning goals; the strategies used to achieve goals; the management of resources; the effort exerted; reactions to external feedback; the products produced [23]. Students' skills in self-regulation entails engaging them in structured, regular diagnostic assessment and self-monitoring, which lead to metacognitive reflection on their learning. It has been found that learners who actively self-regulate achieve higher grades and are more confident than their peers [23]. Metacognition is a term used to describe the various aspects of how a learner processes new knowledge with an explicit understanding and recognition that learning is taking place [26].

Many experiments have been done to find out how accurately students are able to make their self-assessment and accurately

predict different types of grades in different types of situations. Some include forecasting students at the beginning of the course [20], before or after completing the exam, before [26] or after knowing the solutions; other experiences ask students to make their prediction at various times [24] [26] [28]. Calibration is a measure of the degree to which a person's judged ratings of performance correspond to his or her actual performance [26]. Calibration is calculated by taking the difference between a person's self-assessment of performance on a task and his or her actual performance on the task. Self-assessments made prior to performance are called predictions, and those made subsequent to performance are called postdictions. The more closely a person's predicted or postdicted performance matches his or her actual performance (i.e., the difference approaches zero), the better calibrated he or she is [27].

Maybe is important to distinguish between grade aspirations and grade expectations [28] and have clues as to why problems occur, like the lack of an appropriate level of expertise [25] and only because people tend to be overconfident when predicting their performance [30]. In educational settings, many students greatly overestimate how well they will perform on exams. In particular, the lowestperforming students tend to show the greatest overconfidence

(i.e., the "unskilled-and-unaware" effect) [22]. Failures to improve calibration have sometimes been attributed to lack of motivation on the part of participants [27] or the desired levels of performance, wishful thinking [22]. It is suggested that poor performers have more difficulty with metacognitive judgments than their more competent peers' do [30] and poor performers in general are doubly cursed: they lack knowledge of the material, and they lack awareness of the knowledge that they do and do not possess [31], the uskilled and unaware phenomenon. Regarding freshmen, grade inflation in secondary schools may establish unrealistic expectations [32] The characteristics of students who predict grades have been analyzed: Svanunn and Bigatti [33] suggests that students who expected higher grades were somewhat older and reported higher educational goals than those who expected lower grades. Nunn [34] characterizes learning by gender, relating women with academic anxiety. Lundeberg [35] defines that although both men and women were overconfident, undergraduate men were especially overconfident when incorrect. Other authors do not find gender differences in the prediction error [26].

METHODOLOGY

Course Organization

The course is part of a university degree in Computer Science. It is taught in the first semester of the first year and constitutes the students' first contact with computer thinking and a programming language. In this course of propaedeutic nature, a student should, among other skills to be achieved, be able to develop and implement computer solutions for problem solving, that is, to learn correctly and effectively how to program. Before elaborating a program, the student must know how to understand the problem, how to develop strategies for the precise specification of the problem he intends to solve with the machine, establish methods for the detailed and rigorous description of solutions that can be implemented on a computer. The programming language chosen was C. Classroom classes are divided into theoretical and practical laboratory classes, respectively with 2 hours and 4 hours per week.

The evaluation method is based off a continuous evaluation model with four elements of evaluation and attendance requirement above 60%. The tests foresee the use of computers and paper and have an expected duration of 90 minutes with 15 minutes of tolerance. The final grade for the course is determinate as Grade = Test1 * 40% + Test2 * 40% + Project1 * 10% + Project2 * 10%. Considering a 15 weeklong semester, Test1 is the test score taken in the eighth week of classes, Test2 is the test score taken in the last week of classes of the semester. Project1 is the grade given to the student in the project presented in the eighth week of classes and Project2 is the grade assigned to the student in the project presented in the last week of classes in the semester. Both projects were designed with a project based learning methodology: the groups have three elements, they were made up by the teacher and the teacher score corrected by the peer classification.

Survey methodology and data collection

At the beginning of the semester, students were asked to take a survey in order to find out about the demographics of the participants: student identification number, year of birth, sex, previous knowledge of programming languages and whether each student had taken a course at secondary school where computer science was taught. The two surveys that this article uses were requested in the post-test theoretical class, after the date of each test, after the students knew "a" test correction and after the presentation and discussion of each of the projects. The questions for these two surveys n = {1, 2} were:

Id: Student number

PTestn: Forecast test note n

PProjn: Forecast note project n

Selfn: My course self-assessment

The surveys were available on MOODLE, on the course page.

RESULTS

52 students were enrolled and divided into two practical classes. However, 12 students never attended any theoretical or practical classes. 37 students responded to an initial survey: five female (14%) and 32 male (86%). The average age was 19.2 years and the most frequent age was 18 years. The maximum age was 34 and the minimum was 18, with 81% of the students being 18, 19 or 20 years old. 19 students had a computer science course in secondary education: 14 attended computer applications B on the 12th year, four Information and Communication Technologies (ICT) on the 9th year and a Web Design on the 10th, 11th and 12th years. 19 students replied that they had some programming knowledge, having referred to Java, JavaScript, C #, C, Pascal, HTML and CSS, Visual Basic and Python.

30 students took the first test, 13 the second test. Dropouts and absences are not counted. 33 students presented the first project, 12 the second project. We chose to show all the values in graph so that the reading of the data is simple to do. In the following figure we present three graphs: the first test, the first project and the first part. In each of the graphs we present, sorted ascending by grade, the grade that the student actually had in the test (test 1, project 1 and intermediate grade, grade 1, corresponding to 80% of the test and 20% of the project to maintain the proportions of the final semester classification), the student's expected grade and the calibration, delta (difference between reality and the student's expectation). In the chart of test 1 we can see that the delta is negative for worse grades and positive for the best ratings. Looking at the extremes: the student who scored 0.35 predicted to get a grade of 11 (delta = -10.75) and the student who scored 18.9 predicted to have 12 (delta = 6.9). The average of the test grades was of 7.73 and the average of the grades predicted by the students was of 8.03. The delta of grades of up to 6 values was -3.39, from 6 to 12 values it was of -0.07 and in classifications of 12 or more it was of 5.04. In the other two cases, project 1 and mid-semester grade, the differences are not so pronounced but remain high. In project 1, a student who had zero wrote that he thought he would score 14 (delta= 14); on the contrary, two students who scored 18 thought they were going to have 17 as a grade for project 1. The average of the test grades was of 13.41 and the average of grades predicted by the students was of 14.15. The delta of grades up to 6 values was -10.17, from 6 to 12 values was -2.47 and in classifications of 12 or more it was 1.19. Consequently, the evaluation for the middle of the semester takes into account the two previous items: The student with the lowest grade (0.37) predicted that he would score 5, while the student with 18.18, which is the highest grade, predicted that he would have 12. In other words, the delta of the extremes is again negative for the lower grades and positive for the highest grades. The delta of grades up to 6 values was of -5.95, from 6 to 12 values it was of -1.14 and in classifications of 12 or more it was of 1.65.

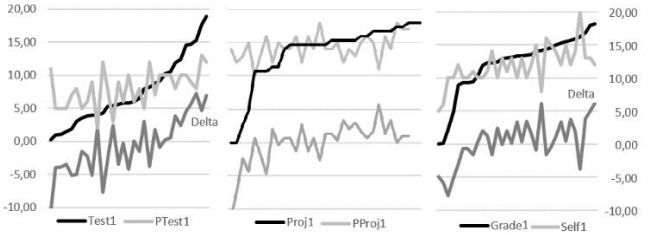

Figure 1. Test1, project1 and mid-semester score: expectation, reality and delta (reality - expected).

The results obtained in the second part of the semester were not as marked as in the first survey. The student who scored 2.2 predicted to get a grade of 2 (delta = 0.2) and the student who scored 16 predicted to have 14 (delta = 2). The average of the test grades was of 6.74 and the average of grades predicted by the students was of 6.69. The delta of grades up to 6 values was of 0.24, from 6 to 12 values it was of -0.48 and in classifications of 12 or more it was of 2. In project 2, a student who had 13 wrote that he thought he would score 13 (delta= 0); on the contrary, the three students who scored 19 thought they were going to have 10, 14 and 16 as a grade for project 2. The average of project 2 grades was of 17.17 and the average of grades predicted by the students was of 12.75. The student with the lowest grade (1.76) evaluation for the end of the semester, predicted that he would have 4, while the student with 16.2, which is the highest grade, predicted to have 16. The delta of grades up to 6 values was -2.24, from 6 to 12 values was -2.34 and in classifications of 12 or more it was 0.2.

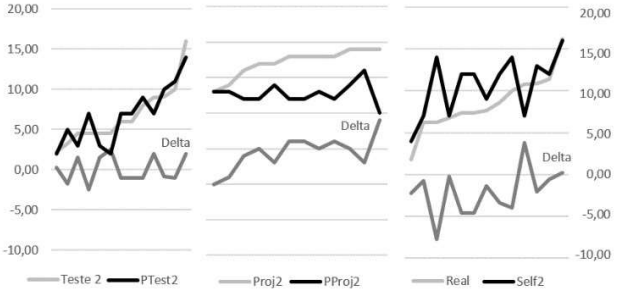

Figure 2. Test2, project2 and final semester score: expectation, reality and delta (reality - expected).

Grades| Delta	Test1	n	Proj1	n	Self 1	n	Test2	n	Proj2	n	Self 2	n
0-6	-3,39	15	-3,34	4	-5,95	4	0,24	6		0	-2,24	1
6-12	-0,08	8	-2,65	5	-1,14	5	-0,48	6		0	-2,34	11
>=12	5,04	7	0,29	24	1,65	24	2,00	1	4,42	12	0,20	1

Figure 3. Average Delta by grades.

We went to do the same exercise for the division by gender, although the women are, as usual in this type of courses, in a large minority. It seems that women have higher selfassessment than men, both in the middle of the semester and at the end of the semester. Men have higher averages for the difference between the grade obtained in project 2 and the reality. In the freshmen and not freshman counter position, there is a difference in the final grade: students who are repeat students get the final grade correct while the freshmen refer a value on average 4 numbers above reality. The reverse happens when we divide students by age: the ones who are over 19 years old have an associated negative 4-point delta, referring to the difference between the final grade and the grade they thought they would get. There are no more significant differences in relation to having (or not having) a computer course in secondary education and having (or not) previous knowledge of programming languages.

Characteristics | Delta	Test 1	Proj1	Self1	Test2	Proj2	Self2
Female	-2,31	-1,80	-5,69	0,20	0,00	-2,24
Male	-0,29	-0,55	-3,20	0,03	4,42	-2,13
Not Freshman	0,22	-0,25	-2,30	-3,38	0,20	0,00
Freshman	-0,86	-0,89	-3,98	-0,65	0,03	4,44

>19 yrs old	0,44 -1,86 -2,74 -1,64 -2,17 -4,94
18 or 19	-1,18 -0,10 -4,05 3,41 2,43 -0,85
K12	0,14 -1,12 -2,01 -0,38 4,78 -2,84
Not K12	-1,11 0,47 -4,73 1,00 2,50 -0,55
PL	-0,59 -1,03 -2,56 -0,05 4,67 -2,37
No PL	-0,60 0,53 -4,16 0,25 2,75 -1,6

Figure 4. Average Delta by sex, be freshman, age, computer science previous, know some programming language.

DISCUSSION

When the students answered the surveys, they already knew what the resolution was. The test is done on the test paper, but most of the resolutions are made on the computer and with the C programming language. However, grade predictions are of particular concern for students with worse grades: they always predict much higher grades. In the case of students with better grades, the opposite is true: they always think that the grade will be worse than it actually is. The case of the first test shows that students with lower grades inflate the forecast, while those with better grades deflate the grade they think they will get. There are many reasons why this could happen: inflated secondary school grades [32], wishful thinking [22], lack of motivation on the part of participants [27] or just the uskilled and unaware phenomenon [31].

The other characteristics (in addition to good or bad grades) do not seem so significant, but we can infer that the biggest differences with the reality of optimistic students appear in test 1 are being a woman (few women in this course), freshman, having 18 or 19 years old, not having had informatics in high school and not having previous knowledge of programming languages. On the contrary in test two, the characteristics will be almost the opposite: being a freshman and being over 19 years old.

If in project 1 almost everyone is too optimistic, the opposite appears in project 2 where men who had a high school computer course and previous knowledge of programming languages have calibrations (delta) of close to 5 values, that is, they predict to have less 5 values of what you have in reality. These are the values that occurred on average in test forecast 1. Regarding the self-assessment, or grade they think they have in the middle or at the end of the semester, they are all too optimistic.

CONCLUSIONS

It is known from literature that there is a great difficulty for students to be able to predict their grades. It has been widely studied and most of the time it is concluded that people in general, and students in particular, are too optimistic. And it can happen for several reasons.

In this study, we assessed the possibility of including selfassessment in an introduction to programming course. There are many studies that show that this is a way to increase students' capacities by integrating them more into the teaching process. We had doubts about the students' selfknowledge and their perception of learning. We noticed that this happens mainly in the first test: the difference between the real and the predicted result is negative for students with grades below 6, being positive for students with grades above 12. Interestingly, it was students with grades between 6 and 12 that best predicted their ratings. We conclude that the more students know, the more they think they do not know and the less they know, the more ignorance they have regarding their knowledge. Thus, it does not seem to us that self-assessment strategy is possible in this way and in this context.

REFERENCES

[1] R. H. Austing , B. H. Barnes , D. T. Bonnette , G. L. Engel and G. Stokes, "Curriculum '78: recommendations for the undergraduate program in computer science— a report of the ACM curriculum committee on computer science," *Communications of the ACM,* vol. v.22 n.3, pp. 147-166, Março 1979.

[2] S. Bergin and R. Reilly, "Programming: Factors that Influence SuccessSusan," in *Proceedings of the 36th SIGCSE Technical Symposium on Computer Science Education*, 2005.

[3] B. A. Becker, G. Glanville, R. Iwashima, C. McDonnell, K. Goslin and C. Mooney, "Effective Compiler Error Message Enhancement for Novice Programming Students," *Computer Science Education,* pp. 148-175, 2016.

[4] A. Robins, J. Rountree and N. Rountree, "Learning and Teaching Programming: A Review," *Computer Science Education,* vol. 13, no. 2, pp. 137-172, 2003.

[5] M. Barak, J. Harward and G. L. S. Kocur, "Reducing the dropout rate in an introductory programming course," *Journal of Science Education and Technology,* vol. 16, no. 4, pp. 71-76, 2007.

[6] M. Barak, J. Harward, G. Kocur and S. Lerman, "Transforming an introductory programming course: From lectures to active learning via wireless laptops," *Journal of Science Education and Technology,* vol. 16, no. 4, 2007.

[7] C. Hundhausen and J. Brown, "What You See Is What You Code: A "live" algorithm development and visualization environment for novice learners," *Journal of Visual Languages and Computing,* vol. 18, no. 1, pp. 22-47, 2007.

[8] G.-M. Kao, S. Lin and C.-T. Sun, "Beyond sharing: Engaging students in cooperative and competitive active learning," *Educational Technology and Society,* vol. 11, no. 3, pp. 82-96, 2008.

[9] C. C. Bonwell and J. A. Eison, "Active Learning: Creating Excitement in the Classroom," *ERIC Digest,* 1991.

[10] P. C. S. E. M. R. W. K. J. S. G. M. &. P. A. Blumenfeld, "Motivating project-based learning: Sustaining the doing, supporting the learning," *Educational Psychologist,* vol. 26, no. 3-4, p. 369–398, 1991.

[11] K. Beck, M. Beedle and A. Bennekum, "Manifesto for Agile Software Development," 2001.

[12] T. Dybå and T. Dingsøyr, "Empirical studies of agile software development: A systematic review," *Information and Software Technology,* vol. 50, no. 9-10, pp. 833-859, 2008.

[13] C. McDowell, L. Werner, H. Bullock and J. Fernald, "Pair programming improves student retention, confidence, and program quality," *Communications of the ACM,* vol. 49, no. 8, pp. 90-95, 2006.

[14] Y. J. a. B. J. W. Dori, "How Does Technology-Enabled Active Learning Affect Undergraduate Students' Understanding of Electromagnetism Concepts?," *The Journal of the Learning Sciences,* vol. 14, no. 2, p. 243–279, 2005.

[15] S. Pedersen and M. Liu, "Teachers' beliefs about issues in the implementation of a student-centered learning environment," *Educational Technology Research and Development,* vol. 51, no. 1, p. 57–76, 2003.

[16] R. Ibrahim, N. Rahim, D. Ten, N. Maarop and S. Yaacob, "Student's opinions on online educational games for learning programming introductory," *International Journal of Advanced Computer Science and Applications,* vol. 9, no. 6, pp. 352-340, 2018.

[17] D. Nulty, "Peer and self-assessment in the first year of university," *Assessment and Evaluation in Higher Education,* vol. 36, no. 5, 2011.

[18] M. Öhrstedt and P. Lindfors, "First-semester students' capacity to predict academic achievement as related to approaches to learning," *Journal of Further and Higher Education,* vol. 43, no. 10, pp. 14201432, 2019.

[19] P. R. Pintrich, "Understanding self-regulated learning," *New directions for teaching and learning,* pp. 97-101, 1995.

[20] V. Prohaska, ""I Know I'll Get an A": Confident Overestimation of Final Course Grades," *Teaching of Psychology,* vol. 21, p. 141–143, 1994.

[21] C. A. Wendorf, "Grade point averages and changes in (great) grade expectations," *Teaching of Psychology,* vol. 29, p. 136–138, 2002.

[22] M. D. K. Serra, "Unskilled and unaware in the classroom: College students' desired grades predict their biased grade predictions," *Memory & Cognition,* vol. 44, p. 1127–1137, 2016.

[23] B. Zimmerman, "Theories of Self-Regulated Learning and academic achievement: an overview and analysis," in *Self-Regulated Learning and Academic Achievement: Theoretical Perspectives*, Taylor & Francis, 2008.

[24] D. J. Nicol and D. Macfarlane-Dick, "Formative assessment and selfregulated learning: a model and seven principles of good feedback practice," *Studies in Higher Education,* vol. 31, no. 2, pp. 199-218, 2006.

[25] R. Belski and I. Belski, "Can students predict their grade accurately in order to selfregulate?," in *Conference of the Australasian Association for Engineering Education - AAEE2013*, 2013.

[26] P. W. Grimes, "The Overconfident Principles of Economics Student:," *The Journal of Economic Education,* vol. 33, no. 1, pp. 15-30, 2002.

[27] D. J. Hacker, L. Bol and K. Bahbahani, "Explaining Calibration in Classroom Contexts: The Effects of Incentives," *Metacognition and Learning,* vol. 3, p. 101–121, 2008.

[28] R. Remedios, D. Lieberman and T. Benton, "The effects of grades on course enjoyment: Did you get the grade you wanted?," *British Journal of Educational Psychology,* vol. 70, pp. 353-368, 2000.

[29] G. Keren, "Calibration and probability judgements: Conceptual and methodological issues," *Acta Psychologica,* vol. 77, no. 3, pp. 217273, 1991.

[30] D. Dunning, K. Johnson, J. Ehrlinger and J. Kruger, "Why people fail to recognize their own incompetence," *Current Directions in Psychological Science,* vol. 12, p. 83–87, 2003.

[31] T. M. Miller and L. Geraci, "Unskilled but aware: Reinterpreting overconfidence in low-performing students.," *Journal of Experimental Psychology: Learning, Memory, and Cognition,* vol. 37, no. 2, p. 502–506, 2011.

[32] R. Landrum, "Student Expectations of Grade Inflation," *Journal of Research and Development in Education,* vol. 32, no. 2, pp. 124-128, 1999.

[33] S. Svanum and s. Bigatti, "Grade Expectations: Informed or Uninformed Optimism, or Both?," *Teaching of Psychology,* vol. 33, no. 1, pp. 14-18, 2006.

[34] G. Nunn, "Adult learners' locus of control, self-evaluation and learning temperament as a function of age and gender," *Journal of Instructional Psychology,* vol. 21, no. 3, p. 260–264, 1994.

[35] M. A. Lundeberg, F. P. W. and J. Punćcohaŕ, "ighly confident but wrong: Gender differences and similarities in confidence judgments," *Journal of Educational Psychology,* vol. 86, no. 1, p. 114–121, 1994.