# PredictCS: Personalizing Programming Learning by Leveraging Learning Analytics

**David Azcona**
Insight Centre for Data Analytics, Dublin City University
David.Azcona@insight-centre.org

**I-Han Hsiao**
School of Computing, Informatics & Decision Systems Engineering, Arizona State University
Sharon.Hsiao@asu.edu

**Alan Smeaton**
Insight Centre for Data Analytics, Dublin City University
Alan.Smeaton@insight-centre.org

**ABSTRACT**: This paper presents a new framework to harness sources of programming learning analytics at a Higher Education Institution and how it has been progressively adopted at the classroom level to improve personalized learning. This new platform, called PredictCS, automatically detects lower-performing or "at-risk" students in computer programming modules and automatically and adaptively sends them feedback. PredictCS embeds multiple predictive models by leveraging multi-modal learning analytics of student data, including student characteristics, prior academic history, logged interactions between students and online resources, and students' progress in programming laboratory work, and their progression from introductory to advanced CS courses. Predictions are generated every week during the semester's classes. In addition, students are flexible to opt-in to receive pseudo real-time personalized feedback, which permits them to be aware of their predicted course performance. The adaptive feedback ranges from programming suggestions from top-performers in the class to resources that are suitable to bridge their programing knowledge gaps.

**Keywords**: Learning Analytics, Machine Learning, Computer Science Education

## 1 INTRODUCTION

PredictCS is a Predictive Analytics platform for Computer Science courses that notifies students based on their performance using past student data and recommends most suitable resources for students to consult. These notifications have been widely adopted since Purdue University launched Course Signals (Arnold & Pistilli, 2012). Dublin City University's PredictED project emulated Purdue's and notified students their rank within the class (Corrigan & Smeaton, 2015). Both systems yielded impressive improvement in first-year retention rates.

Recently, researchers have been working on augmenting the Integrated Development Environment (IDE) or programming environment by crowdsourcing code solutions. Students are suggested error corrections or solutions that peers have applied before. Java has been the programming language targeted the most with platforms such as University of Durham's BlueFix (Watson & Godwin, 2012). In addition, also using code snapshots, Carnigie Mellon's ITAP (Intelligent Teaching Assistant for

Programming) provides hints as feedback while programming and has been implemented in CloudCoder (Rivers & Koedinger, 2017).

In Dublin City University, Dr. Stephen Blott, lecturer at the School of Computing, has developed a Virtual Learning Environment (VLE) for the teaching of computer programming. The system includes a grading platform that provides real-time feedback on each programming submission by running a suite of test cases. This system gives no code suggestions or personalized help for errors.

# 2 INFRASTRUCTURE

## 2.1 Development of the platform

PredictCS has been designed to enhance and personalize the student's learning in programming modules. The platform sits on top of Dr. Blott’s VLE grading system and leverages that submission data. Figure 1 shows how PredictCS is inputted a combination of student characteristics, past academic results, programming submissions and interactions with the material to generate predictions and recommendations to the students, lecturers and Faculty.

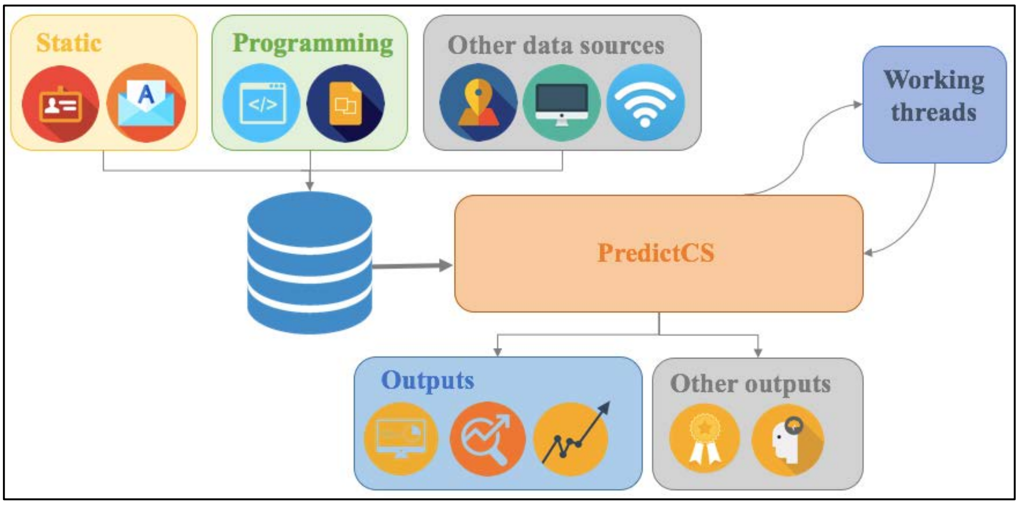

**Figure 1: Infrastructure of PredictCS**

Our contribution is to combine the detection of students “at-risk” in programming modules and the feedback we provide students with timely weekly notifications by leveraging Learning Analytics approaches jointly with suitable code solutions and resources. Figure 2 shows how the combination of Learning Analytics with programming data enables us to provide students with personalized feedback in Computer Science courses. The feedback contains a message based on their predicted performance, recommended resources and programming code solutions from top-students in the class. Code solutions are found to have a positive effect on the student’s learning (Nguyen et al, 2016). We work with submissions and typically short programs so the feedback given to students can be on submission at the platform or via email and we do not have snapshots of students developing the code to learn from their design. We are focusing on email notifications but it can change in the future.

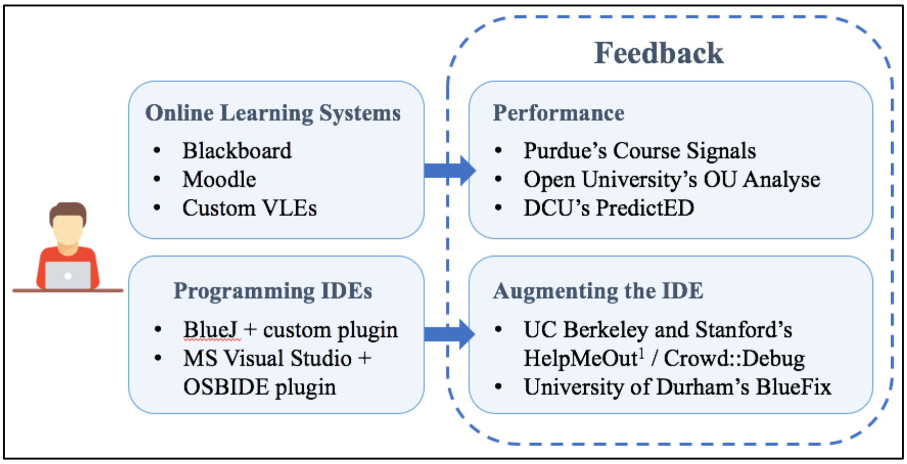

**Figure 2: PredictCS combines Learning Analytics with programming data**

## 2.2 Innovation

The platform's goal is to enhance and motivate students to learn programming skills. We apply Learning Analytics as we gather significant data about students learning to program and interacting with the material. Using that digital footprint allows us to understand better their knowledge gaps and help them learn more. We do so by sending notifications about performance, suggested programs and material on a weekly basis to students that opt-in. We developed a recommendation engine as a module for PredictCS that suggests code solutions from top-students in the class as seen in Figure 3. In addition, specific material lessons are recommended to students on these notifications.

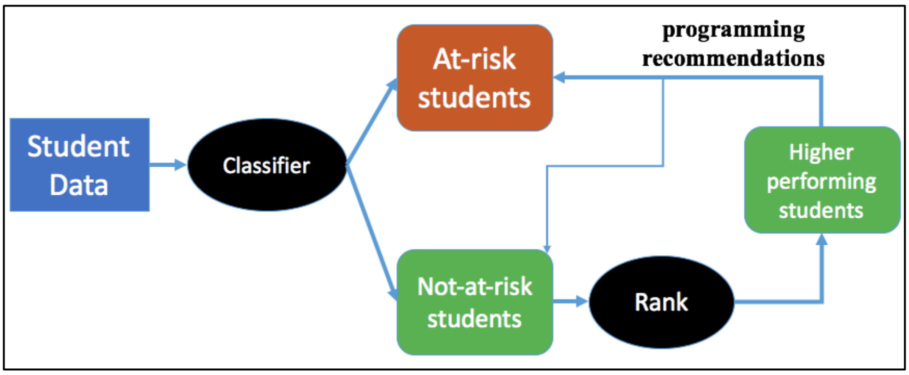

**Figure 3: PredictCS Recommendation Engine**

The main benefits of the innovation are to help navigate students to the areas they should focus on for their learning progression and, at the same time, notify lecturers where the class as a whole is struggling in order for them to adapt their teaching and curriculum. This can be achieved by using data gathered from past student cohorts and applying machine learning and collaborative filtering techniques.

The ideal scenario for teaching and learning practices would be to design and plan material based on the student's progress and learning using our platform, where lecturers can detect and attend to disengaged students, see how the learning of concepts get propagated through the class and seeing how feedback helps their learning and motivation. We found face-to-face discussions with researchers on the project help lecturers make the most of the platform.

# 3 STAKEHOLDERS

## 3.1 Teachers

As a prerequisite for the use of PredictCS, teachers have to utilize the custom grading VLE for the teaching of programming. That requires the teachers to upload the laboratory exercises and the corresponding unit test cases to a server for the grading of them. That way, we gather data about the programming submissions from students. Lecturers should be comfortable uploading and checking material using web technologies, have a basic understanding of probabilities and how PredictCS perform predictions and recommendations. Believing data-driven technologies can enhance and help their teaching is desirable.

PredictCS supports the preparation of teaching activities by Informing lecturers where students are having more issues (i.e. which programming exercises students fail the most) and how students are likely to perform on the next assessment. In short, a report is also sent to lecturers with top issues of the week (the exercises students have the most difficulty) and ranked list of students and their performance likelihood of failing the next assessment. The system sends weekly notifications to lecturers containing who may be at-risk and what are the difficulties students find, but unfortunately, we are not able to measure whether teachers read these emails thoroughly.

PredictCS gives them insights now about how students are doing in the laboratory sessions. Lecturers try to adapt based on the students' needs and issues with the material. They are now better aware of their students' progress by looking at PredictCS and, also, their results at laboratory examinations. The biggest challenge for the lecturers is to find ways to reach as many students as possible in a personalized way. Automatic grading and data analytics tools are helping us to move forward in that space.

Teachers were interviewed at the end of each semester and admitted with large class sizes for programming modules it is practically impossible for them to monitor each student personally and these automated approaches were useful. Simply seeing the list of students marked red or green each week gave them a sense for how things were going. However, they pointed out the predictions seemed too negative as researchers were trying to maximize the students marked "at-risk" while keeping a good balance between the two classes.

## 3.2 Students

In order to receive personalized feedback, students need to opt-in to the notifications. After their first laboratory exam, a pop-up will appear on the VLE's submission platform to opt-in or out and where they can find further information about the project. Then, the system sends customized notifications every week based on the progress, suggested resources to focus their learning and programming code solutions to the students that opted-in. Every notification contains an unsubscribe link for them to opt-out at any time. Students do not need any pre-requisites but a positive attitude towards problem solving for them to stay motivated.

Researchers gathered the students' opinions about the platform and the notifications via a written questionnaire. Most students would recommend the system to students attending the same module next year or would like to see it included in other modules. Students who were doing very well were getting a similar response every week and were demanding more personalized feedback.

### 3.3 Researchers

Learning to program involves a variety of complex cognitive activities, from conceptual knowledge construction to basic structural operations. This discipline is very logical and some students find it easier than others in the beginning. Researchers on this project strive to provide the tools for lecturers to adapt their teaching based on their classes' progression and for students to stay engaged and focus where they need to. A web application is also maintained by these researchers where the analysis of the machine learning classifiers, the predictions for every week and all the notifications sent to students can be found. Lecturers and Faculty have access to this platform.

### 3.4 Faculty

Faculty at Dublin City University's School of Computing have been running research projects around student retention and engagement through Learning Analytics for the past few years. It took them months to prepare a proposal for the University Research Ethics Committee. Now, researchers can request access to this programming related data and static information about students. The University Research Ethics Committee and the data commissioner will review these applications and grant access if appropriate. The school administration owns the data generated.

Dublin City University is partnering with other Higher Education Institutions such as Arizona State University to share methodologies and pioneer approaches around Learning Analytics and Computer Science Education.

## 4 ADOPTION

In Computer Science Education, lecturers teach programming courses that contain a considerable amount of laboratory work for students to learn these skills. That allows data-driven technologies such as our platform to perform well.

PredictCS is an ongoing project and has been in place for more than three semesters in our university and ran in more than six computer programming modules. At first, we ran a retrospective analysis and developed predictions for student performance using cross-validation techniques on past student data that confirmed the potential of these systems. The following semester we ran pseudo-real time weekly predictions using only one year of training data and targeted a new cohort of students individually during laboratory sessions (Azcona & Smeaton, 2017). Then, the next year, we sent notifications to students based on their performance and suggested resources and have been increasingly adding code solutions, suggested reading material and laboratory sessions to focus their attention on.

Students that corrected their programs and re-submitted them based on the suggestions improved their performance with respect to the ones that did not. In addition, the differential improvement between the two laboratory exams also increased the year the predictions were run and the notifications were sent compared to the previous academic year that our models are trained with.

In terms of the ethical issues, some students may want the data generated by their learning to be deleted and they can request the university to do so. However, that's the way we measure their progress and performance.

## 5 CONCLUSION

This study has proven to successfully create a programming digital footprint we can leverage. Our data approaches to select a learning algorithm or a subset of predictors on data models enabled us to identify students having issues in programming courses and to provide them with timely interventions. This system and these notifications have shown promising results and will be implemented in more programming modules so we can have a broader impact. We will also be more vocal about the project so more students can opt-in, understand and benefit from this research. We are eager to provide our students with more detailed programming recommendations, suitable material and other actions to fill the knowledge programming holes they may have while learning Computer Science programming design at our university so we are working towards improving some modules of the platform.

## REFERENCES

Arnold, K. E., & Pistilli, M. D. (2012, April). Course signals at Purdue: Using learning analytics to increase student success. In *Proceedings of the 2nd International Conference on Learning Analytics and Knowledge* (pp. 267-270). ACM.

Azcona, D., & Smeaton, A. F. (2017, September). Targeting At-risk Students Using Engagement and Effort Predictors in an Introductory Computer Programming Course. In *European Conference on Technology Enhanced Learning* (pp. 361-366). Springer, Cham.

Corrigan, O., Smeaton, A. F., Glynn, M., & Smyth, S. (2015). Using educational analytics to improve test performance. In *Design for Teaching and Learning in a Networked World* (pp. 42-55). Springer, Cham.

Nguyen, Q., Tempelaar, D. T., Rienties, B., & Giesbers, B. (2016). What learning analytics-based prediction models tell us about feedback preferences of students. *Quarterly Review of Distance Education, 17*(3), 13.

Rivers, K., & Koedinger, K. R. (2017). Data-driven hint generation in vast solution spaces: a self-improving python programming tutor. *International Journal of Artificial Intelligence in Education, 27*(1), 37-64.

Watson, C., Li, F. W., & Godwin, J. L. (2012, September). BlueFix: using crowd-sourced feedback to support programming students in error diagnosis and repair. In *International Conference on Web-Based Learning* (pp. 228-239). Springer, Berlin, Heidelberg.