Prediction of Student Final Exam Performance in an Introductory Programming Course: Development and Validation of the Use of a Support Vector Machine-Regression Model

Veerasamy Ashok Kumar[1*], Daryl D'Souza[2], Rolf Lindén[1] and Mikko-Jussi Laakso[1]

[1] University of Turku
Turku, Finland

[2] RMIT University
Melbourne, Australia

**Corresponding author's email: askuve [AT] utu.fi*

ABSTRACT—*This paper presents a Support Vector Machine predictive model to determine if prior programming knowledge and completion of in-class and take home formative assessment tasks might be suitable predictors of examination performance. Student data from the academic years 2012 - 2016 for an introductory programming course was captured via ViLLE e-learning tool for analysis. The results revealed that student prior programming knowledge and assessment scores captured in a predictive model, is a good fit of the data. However, while overall success of the model is significant, predictions on identifying at-risk students is neither high nor low and that persuaded us to include two more research questions. However, our preliminary post analysis on these test results show that on average students who secured less than 70% in formative assessment scores with little or basic prior programming knowledge in programming may fail in the final programming exam and increase the prediction accuracy in identifying at-risk students from 46% to nearly 63%. Hence, these results provide immediate information for programming course instructors and students to enhance teaching and learning process.*

Keywords—Prior programming knowledge; At-risk students; Predictive data mining models; Machine learning techniques

1. INTRODUCTION

The use of educational technologies to support student learning has offered opportunities to capture vast amounts of information about events related to student engagement and demands a pressing need for data mining in educational research to gain insight into teaching and learning. For example, predicting student performance using predictor variables extracted from learning management system has been important research and there have been numerous efforts to improve student performance (Conjin;Snijders;& Kleingeld, 2017). However, improving student performance is a big challenge at universities. One possible way to improve student academic performance is to predict student performance. By using predictive modelling it is possible to predict student performance in a course and identify students at risk of course failure. Furthermore, predictive models can be used as an early warning system to identify students who need support in a course; and instructors may then use a variety of strategies to communicate with those at-risk students and provide them pathways for improving their performance in the course (Krumm, Waddington, Teasley, & Lonn, 2014). The challenge in establishing a valid predictive model is to judiciously choose a number of independent variables that influence an outcome. Hence, it is important to identify significant predictor variables for a predictive model to get expected results in order to draw sense making decisions. In addition, as Pardo (Pardo, 2014) reported, designing the most suitable model for predictive analytics is important. However, a suitable statistical techniques and methods are required to design those kinds of models. Keeping these points in mind, the objective of the research reported in this paper is to develop a predictive model with selected predictor variables using data mining techniques to predict student performance and at-risk students in a programming course (at university level) to make proactive measures in teaching and learning.

Computer programming is a fundamental course to Computer Science and IT curricula which covers programming concepts that are essential for subsequent advanced programming courses. At our university, this course is also offered to students who are from other disciplines. Typically, students perform poorly in or fail introductory computer programming (Watson;Li;& Godwin, 2014; Marios Tacio Silva, 2014; Zingaro, 2015). Moreover, programming is one of the toughest

courses for novices and the difficulty of learning to program also dissuades students from other programs to avoid taking programming and to select alternative courses (de-la-Fuente-Valentín, Pardo, & Kloos, 2013; Ali & Smith, 2014). Many teaching techniques and educational technologies have been developed by researchers to improve student learning outcomes in programming (Borra & Ciaccio, 2010). However, educators continue to look for factors that impact student performance in programming in order to alleviate student programming learning skills, in turn, to reduce course failure and dropout rates (Asif, Merceron, & Pathan, 2015; Bergin & Reilly, 2005). Furthermore, many academic researchers and educators have made systematic attempts to identify significant predictor variables to construct effective mathematical models to predict student academic performance (Abu-Oda & El-Halees, 2015; Guo, Zhang, Xu, Shi, & Yang, 2015; Longi, 2016) in order to identify students at-risk of course failure. However, the mixture of such predictor variables varies from one academic environment to another due to different cohorts of student, cultural setting and the education systems. Moreover, many of those studies need further verification as the factors identified and results obtained were inconsistent (Longi, 2016). So, the objective of this research is to identify significant predictor/independent variables that predict student programming final exam performance with the aim of identifying students at-risk of failing. Moreover, identifying attributes that affect student's academic performance, and predicting low performance students in programming course has been a long-standing problem. So, it is important to develop an effective model to predict student academic progress, and factors that influence student academic performance in learning programming. Similarly, assessing student academic performance in programming course is important because it helps instructors, course administrators, and academic advisors to take necessary steps in order to improve student learning. Notably, the results emerged from this analysis can be used to identify student achievement and retention status to help instructors to reshape pedagogical practices in programming. As such, this study attempted to develop a model that uses data mining techniques to predict student academic performance in programming courses.

This study used the data that collected and captured via ViLLE, an e-learning tool and deployed Support vector machine–regression learning algorithm to design our model for prediction analysis. Support vector machine (SVM) is a supervised and popular machine learning technique, which contains a set of supervised learning methods that can be used for classification, regression and outliers detection. It makes predictions about future instances based on externally supplied instances called input set. Moreover, SVM performs well on small databases with less generalization error (Hämäläinen & Vinni, 2006). Additionally, K-fold cross validation has been used to evaluate the performance of our predictive model due to the following reasons. If the predictive performance of the model on the training dataset is good but has poor performance or has failed to predict anything useful on the test or unseen dataset then this situation is called overfitting. The overfitted model has poor predictive performance and cannot be generalized. Therefore, it is important to apply K-fold cross validation to report the mean square error and root mean square error to estimate the prediction accuracy of the model.

Consequently, this study developed a validated set of mathematical model based on SVM-regression algorithm with K-fold cross validation procedures to predict student academic performance in programming course. As noted, the main objective of this study is to provide a valid mathematical model with selected factors to predict student final programming exam scores, and at-risk students who may face academic difficulty at early stage of the course to help them to succeed. Towards this objective, this paper addresses the following research questions.

Research questions

- What is the optimal combination of predictor/independent variables with the highest prediction accuracy for predicting student's academic performance?
- What is the percentage of academically at-risk students that can be correctly identified by the model?

In order to find answers to the research questions we collected and analyzed student assessment in the course Introduction to Programming using students' web based homework and demo exercises, prior programming knowledge, and final exam data, to develop a mathematical predictive model. This paper is further divided into six sections. Section 2 presents a literature review of studies involving educational data mining techniques that adopted for this study, the significance of assessment tasks as predictor variables, such as homework exercises, class demonstration activities, and prior programming knowledge on student achievement in final programming examination. In addition, it discusses the role of data mining techniques, SVM algorithm in predictive modelling in details. Section 3 presents the research methodology with details of assessment tasks that were chosen for analysis. Section 4 presents the research outcomes with statistical results. Section 5 discusses these results. Finally sections 6 and 7 present the conclusions and future study and educational implications.

2. LITERATURE REVIEW

2.1 Identifying predictors of student achievement

Predicting student academic achievement is more challenging than teaching since student academic performance affected by various factors (Longi, 2016; Evans & Simkin, 1989). So, identifying and selecting suitable predictor variable(s) with mathematical approaches is important when defining a mathematical predictive model with factors that influence students' academic performance. Student academic performance may be affected by various factors. Evans et al. (Evans & Simkin, 1989) listed 34 independent variables that might be used to predict programming aptitude of students. All those listed variables were categorized into demographic, academic, prior computer training or experience, or behavioral types. Similarly, Austin (Astin, 1978) has listed 146 characteristics as significant data points/factors that can be used to predict student learning outcomes. In addition, Austin identified 192 variables that are associated with institutional and student environment for academic analysis models. However, it seems, the interrelationship between student demographic, personal, educational background, psychological and academic progress data are so complex that these variables correlated in a complicated nonlinear way (Guo, Zhang, Xu, Shi, & Yang, 2015). Consequently, researchers are searching for valuable predictors that can produce consistent results on predicting student academic achievement despite contextual issues that impact student performance. So, the predictor variables routed based on Educational psychology and Bloom's taxonomy was chosen to predict student performance in programming exam despite contextual factors that affect student achievement for our model. For example, prior knowledge in programming is often discussed and included in the predictive models as an input variable (Longi, 2016; Grover, Pea, & Cooper, 2016).

2.2 Prior knowledge as a predictor of student performance

Prior knowledge generally refers to information that the learner or student already learned or acquired through past experiences. Prior knowledge can be defined as a combination of knowledge and skills and it has received considerable attention in educational and psychological research studies (Hailikari, 2009; Ausubel, Novak, & Hanesian, 1978) noted that prior knowledge is the single most important factor in learning, which brings all of educational psychology into just one principle. As noted, many research studies acknowledge that prior knowledge plays an important role in learning and stands out as a significant variable affecting and predicting student academic achievement in several courses (Kebritchi;Hirumi;& Bai, 2010; Hailikari, 2009). Although a few other studies claim that prior knowledge may raise student misconceptions in learning (Alexandron, Armoni, Gordon, & Harel, 2012; Öncu, Sengel, & Delialioglu, 2008), many studies acknowledge that prior knowledge is a strong predictor of student academic achievement (Hailikari, 2009; Holden & Weeden, 2003).

Topic-specific knowledge is knowledge that is specific to the topic or domain, such as programming. There have been several studies conducted on the impact of student prior knowledge about the topic in various courses, including programming (Seery, 2009; Veerasamy, et al., 2018). For example, prior knowledge of programming languages or prior programming knowledge has a significant impact on student success in their first university programming (Rosenschein, Vilner, & Zur, 2004). Prior programming knowledge refers to knowledge and skills that student has acquired from previous programming courses or via their self-directed learning. Almost all academic studies have acknowledged that prior programming knowledge has a marked effect on student learning outcomes and academic performance in programming (Hsu & Plunkett, 2016; Watson;Li;& Godwin, 2014; Hsu & Plunkett, 2016). In addition, Wong (Wong, 2014) confirmed that students who had prior knowledge in programming achieved higher grades in their non-programming courses than those who had no prior programming knowledge (Wong, 2014). However, Kinnunen et al. (Kinnunen & McCartney, 2007) reported that students with acquired programming ability and skills through non-object oriented programming courses did not gain a deep understanding in their learning of object-oriented programming courses. Prior programming knowledge, on the other hand, Tafliovich, et al. (Tafliovich, Campbell, & Petersen, 2013) reported that most of the students with prior programming knowledge acquired via previous programming courses or self-initiated programming and those without prior programming knowledge felt prior knowledge in programming would help succeeding in programming courses. Furthermore, Byrne et al. (Byrne & Lyons, 2001) confirmed that both male and female students who had similar prior programming knowledge secured similar grades in the programming course. Although a few other studies claim that inaccurate prior programming knowledge may hinder new learning and raise misconceptions (Holden & Weeden, 2003; Byrne & Lyons, 2001), many other studies confirm that prior programming knowledge may serve as a reliable predictor of student achievement (Longi, 2016; Hsu & Plunkett, 2016; Watson;Li;& Godwin, 2014). However, the academic performance of the student can be predicted by various factors other than student prior knowledge (Longi, 2016). For example, instructors often use formative assessment task results as an evaluation tool to monitor the level of student progress in learning and to partially determine the student final performance. In addition, formative assessment is transmitted based on Bloom's taxonomy concept (S. & Hastings, 1971).

2.3 Formative assessment tasks as predictors of student performance

Formative assessment is the process of evaluating a learner's knowledge, skills, and attitudes, which helps instructors to provide ongoing feedback to students to improve their learning skills. Moreover, these types of assessment help students to monitor their performance and progress (Corbett & Anderson, 2001). The results emerged from these assessments signal instructors to identify students who need attention to focus on related supporting activities. For example, homework is one of the traditional and continuous summative assessments that instructors use to evaluate student performance. Notably, students who achieve high score in homework assessments perform significantly better in the final exam than those who do not (Veerasamy;D'Souza;Lindén;Kaila;Laakso;& Salakoski, 2016). In addition, homework is an important determinant of student final exam marks (Veerasamy;D'Souza;Lindén;Kaila;Laakso;& Salakoski, 2016; de-la-Fuente-Valentín, Pardo, & Kloos, 2013). Similarly, a review of literature review related to homework in programming courses indicates that homework has a positive effect on student learning outcomes and can be considered as the significant predictor of learning achievement (Su, Huang, Yang, Ding, & Hsieh, 2015). Moreover, e-Homework assignments in programming courses assist students to review their homework solutions to improve their programming learning skills and which impact their summative assessments performance (Vogel-Heuser, Rehberger, Frank, & Aicher, 2014). Apart from the above, assessment tasks such as assignments, lab exercises, and lab exams are used in programming courses to evaluate student coding skills for grading. However, a few other studies claim that formative assessment feedback did not help novice programming students in terms of improving their academic performance (Koulouri, Lauria, & Macredie, 2015; Lin & Chen, 2006).

2.4 Predictive data mining model for students' performance

2.4.1 Predictive data mining model for student performance in computing education. Predictive modeling is a process that uses statistics with collected data and relevant predictor variables to predict future results. It is a kind of mathematical model and may employ classifiers or regressors to formulate a statistical model. Notably, there have been studies of predictive modelling employed both traditional and modern data-mining techniques (Huang & Fang, 2013). Several studies have been conducted to construct predictive data mining models for predicting student performance (Bergin, Mooney, Ghent, & Quille, 2015; EIGamal, 2013). In most of these studies, researchers have employed linear regression, artificial neural network, Naïve Bayes, and SVM learning methods for developing predictive models. In addition, several studies have reported mixed results on using data mining techniques for comparison and prediction. For example, Devasia, et al. (Devasia, P, & Hegde, 2016) reported that Naïve Bayesian's prediction for identifying final grades of computer science students is more accurate as compared to other data mining methods, including linear regression, decision tree, and neural networks. However, Bergin, et al. (Bergin, Mooney, Ghent, & Quille, 2015) reported that there were no statistical differences between the prediction accuracy of Naïve Bayes, logistic regression, support vector machine, artificial neural network and decision tree data mining techniques, in predicting introductory programming student performance. Another study reported that linear SVM is one of the fast machine learning algorithms that widely used in predictive modelling for classification and regression analysis (Kamruzzaman, Sarker, & Ahmad, 2003).

2.4.2 Predictive modelling with SVM. SVM are supervised learning models and considered as a powerful instrument for predicting student performance; its use has demonstrated better performance over other methods (Huang & Fang, 2013; Huang, 2011). The recent version of SVM for regression was developed based on structural risk minimization, which minimizes the errors in training data of the model. Consequently, this feature of SVM enables it to provide more accurate prediction and to increase the generalizability of the model (Vapnick, 1995). For example, Huang et al. (Huang & Fang, 2013) reported SVM models having the highest percentage of accurate predictions when compared with other mathematical models, such as, multiple linear regression, multiple layer perception, and radial basis function (Huang & Fang, 2013). Lee (Lee, 2016) conducted a study to predict students problem solving performance using SVM, and concluded that SVM based predictive model provided better predictions on problem solving performance of students. However, Marbouti, et al. (Marbouti, Diefes-Dux, & Madhavan, 2016) conducted a similar comparative study to identify at-risk students for first year engineering students and concluded with mixed results. Marbouti et al. (Marbouti, Diefes-Dux, & Madhavan, 2016) identified that the SVM's prediction for identifying at-risk students is less accurate as compared to the Naïve Bayes classifier, although overall prediction accuracy of both methods for identifying students who passed the course was similar. However, Costa, et al. (Costa, Fonseca, Santana, & de, 2017) conducted a study on evaluating the effectiveness of various educational data mining techniques for predicting student at-risk at the early stages of introductory programming courses and confirmed that SVM is sufficiently effective to identity students at-risk of academic failures in programming courses. Based on these research studies we surmise that the SVM-regression algorithm is generally good for numerical prediction and has a high generalization performance.

Although there have been many studies conducted on identifying the number of factors and mathematical models that predict academic performance of students at school, college and university level, the aforementioned studies did not confirm that including more predictor variables would improve the prediction accuracy of any predictive model (Yuer &

Güngörmüş, 2011; Vihavainen, 2013). Additionally, as reported, most predictive models include two or more predictor variables to bring the possibility of multicollinearity (Dickey, U., & Raleigh, 2012). The multicollinearity model (a model with predictor variables that are correlated with other predictor variables) may raise inconsistent results and prediction accuracy, which forces it to assess the selection of predictor variables on predicted variables. Notably, only a few research studies included several variables to develop predictive models in order to predict student performance in programming courses (Romero, López, Luna, & Ventura, 2013; Vihavainen, 2013). In addition, although SVM based research studies conducted on predicting student academic performance for various courses, relatively a few SVM-regression based research studies only conducted to identify possible predictor variables, and to predict programming students' academic performance. As such, this study deployed SVM-regression based predictive model that explain data with a minimum number of predictor variables to predict students' final programming exam performance.

3. RESEARCH METHOD

The primary purpose of this study was to predict student final programming exam performance including students at risk in a programming course by developing a valid predictive model. Hence, in this study we define a predictive model that contains predictor variables; homework, demo and prior programming knowledge levels to predict the student final programming exam results and to identify students at risk of failure in a programming course. This study used SVM-regression modelling to develop a predictive model based on the dataset collected from 190 novice programming students in an introductory programming course over five spring semesters (2012, 2013, 2014, 2015 and 2016). The objective of this SVM-regression model is to predict the final programming exam performance of individual students to identify at-risk students from those predicted scores. The student homework and demo exercise scores, along with the student prior programming knowledge are used to predict student final exam performance. This study used supervised machine learning technique, in which the training data set is provided with the corresponding target. The predicted target space is continuous for regression problems. Additionally, the objective of building the prediction model is to design a model that most precisely estimates the desired output value for new data (Fortmann-Roe, 2012). To achieve this goal, it is important to obtain the estimate of model's prediction error to evaluate the performance of fitted model. So, in order to evaluate the performance of our model this study used K-fold cross-validation technique. K-fold cross-validation is a model evaluation method that is widely used to estimate the prediction error of observed values. Cross validation is a standard model evaluation method which provides a nearly unbiased but highly variable estimator (Kim, 2009). The goal of the cross validation is to define the dataset to "train" and for "test" in order to avoid overfitting model selection. Then, it also assesses how well a model generalizes to unknown dataset. In K-fold cross-validation approach, the data set is divided into k subsets randomly. Then the cross validation on model building and error estimation is repeated K times. Each time, one of the k-subsets is used as "test set" and the other k-1 subsets are put together to form a "training set". Finally, the average of recorded prediction errors across all K trials is computed to obtain the performance metric for the predictive model. So, this study used K-fold cross-validation method to evaluate the predictive model performance. This study applied 5-fold cross validation method to get the true error estimation for the defined predictive model. Over the five years there were 291 students enrolled for this course. Of these, only 190 students attended the final exam and therefore the study is limited to the academic data of these 190 students. So, in total 190 student academic data used for supervised machine learning including 5-fold (K=5) cross-validation method. R software was used to deploy a model with 5-fold cross-validation approach to predict the student final exam scores. Thus, the output of this model is the students' scores in the final programming exam.

3.1 Description of the course and data allocation

This backdrop for this study was the course entitled Introduction to Programming, which used Python as the vehicle of instruction. The results were used for homework, demo, final exam, and prior programming knowledge for the years 2012, 2013, 2014, 2015 and 2016 to validate the predictive model against defined research questions. Introduction to Programming is designed for students with no prior programming knowledge. This course is offered once a year and the duration of the course spans a semester of 12 weeks. The course uses ViLLE (Rajala & Erkki Kaila, 2005) as learning management system to support technology-enhanced classes. This software is mainly used to deliver and manage course content, such as lecture notes, formative and summative assessment tasks for programming students. Student academic data was able to be collected via ViLLE and used SPSS and R software for statistical analysis. The breakdown of the 190 students, who took the course and attended e-final exam over the five years, was as follows: 27 students in year 2012, 42 students in year 2013, 42 students in year 2014, 18 students in year 2015, and 61 students in 2016.

3.2 Description of predictor variables

The assessment tasks for Introduction to Programing are delivered and graded by ViLLE. These assessment tasks include homework and demo exercises, and an online final examination. The submitted assessment tasks are recorded via

ViLLE to calculate final grades (Veerasamy;D'Souza;Lindén;Kaila;Laakso;& Salakoski, 2016). To attain a pass in the course, students are required to pass both the formative and summative assessments listed below. The assessments are further described below.

3.2.1 Homework exercises (HE): This represent weekly formative assessment tasks distributed via ViLLE for students to practice and submit their answers electronically over a total of 10 weeks. Student must obtain at least 50% over all in the weekly tasks in order to pass this component and the course.

3.2.2 Demo exercises (DE): As per DE these are weekly tasks conducted after lectures over 10 weeks. Programming exercises are provided via ViLLE for students to complete before attending the DE session. A few students then randomly selected via ViLLE to demonstrate their answers in the supervised classes. No marks will be awarded for class demonstrations. Students who complete DEs are asked to submit all completed exercises via ViLLE enabled in lecturer's computer. The DE is also a hurdle and student must obtain at least 40% over all in the demo tasks in order to pass this component and the course.

3.2.3 Final Exam (FE): This is an online summative assessment conducted at the end of the course of study. This FE is conducted electronically using ViLLE. The FE is a hurdle and student should secure at least 50% marks to pass the hurdle and the course.

3.2.4 Prior programming knowledge (PPK): The course entry survey is included as a part of the course. ViLLE was used to create and conduct survey, which was conducted at the beginning of the semester and for which the response data was stored in ViLLE. The survey questions were prepared in English. The purpose of this survey was to capture the learning profile of the students, specifically the details of their prior programming knowledge (using a Likert scale) and the preferred learning style (choosing one option from among multiple choices). This survey data was collected during a very early stage of the course. To establish prior programming knowledge, a three point Likert scale was created. This three-point rating was designed to make sure that the prior programming knowledge survey question has an optimum number of response categories and a number beyond which there is no further improvement in discrimination between the rated items (Jacoby & Matell, 1971). Moreover, each point was presented to students with clear description to accurately self-assess their prior knowledge of programming (Table 1) (Veerasamy, Daryl D'Souza, & Laakso, 2018).

Table 1: Survey question to examine student's prior programming knowledge - self reported

Question: How much previous programming experience/knowledge (PPK) have you had?	
PPK level	Meaning
0	Means you have no programming experience / knowledge at all.
1	Means you have learnt or acquired some basic skills in programming. In addition, you may know how to write and execute basic level computer programs.
2	Means you have studied one or more programming languages or you have sufficient knowledge in programming. In addition, you know how to write mid-level and or higher level computer programs.

3.3 Hypothetical predictive model development

The main objective of this study was to develop a predictive model that can help the lecturer to predict student final programming exam scores so as to support the facilitation of proactive measures to improve student outcomes. Based on the research findings reported, performance in the course is hypothesized to be a function of the following student data: (1) prior programming knowledge level, (2) homework scores and (3) demo exercise scores. Specifically, the results of Kruskal-Wallis and Bonferroni post-hoc tests of our previous studies has been used to determine if there were statistically significant differences between student prior programming knowledge levels, based on the dependent variable; final exam performance (Veerasamy, Daryl D'Souza, & Laakso, 2018). In addition, the multiple linear regression analysis has been used to assess these selected predictor variables to explain the criterion (dependent) variable. Subsequently these identified variables were used for predictive model development. Table 2 shows the relationship between the independent (predictor) variables and dependent variable (predicted) as results of regression analysis to develop our hypothesized predictive model.

Table 2: Multiple linear regression results –relationship between HE, DE and PPK on FE

Independent variable	R 0.525	Adjusted R Square 0.264	p-value	Coefficient
Homework exercises (HE)			`	0.066
Demo exercises (DE)			0.002	0.034
Prior programming knowledge (PPK)			0.000	7.473

The coefficients for HE (0.066), DE (0.034), and PPK (7.473) are significantly different from 0 because their p-values are 0.000, 0.002, and 0.000, and notably those values are smaller than 0.050. These results show that there is a significant relationship between prior programming knowledge levels and the selected assessment tasks scores on final exam performance. In addition, these results reveal that PPK, HE and DE may statistically significantly predict final exam scores. Figure 1 presents the hypothetical predictive model based on the review of prior knowledge and formative assessment tasks in student achievement literature, multiple regression results, and the experience in teaching programming courses.

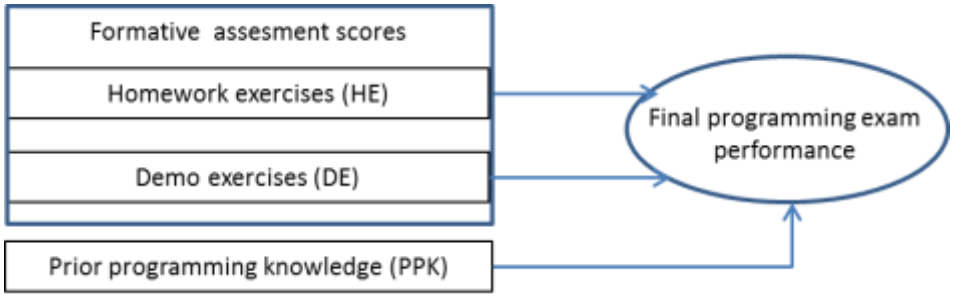

Figure 1: Hypothetical predictive model of the study

Thus, in this study we hypothesized that formative assessments tasks such as homework and demo, and prior knowledge in programming are connected, and these all influence student final programming exam performance. Thus, PPK, HE and DE scores are categorized as input variables.

4. DATA ANALYSIS AND RESULTS

The full dataset collected in years 2012, 2013, 2014, 2015, and 2016 (n=190) was used to develop and to validate a predictive mathematical model. This study used the SPSS and R software to store and analyzes the data. The collected dataset was preprocessed. The data pre-processing was done in the following ways; first, the numerical values of all data were scaled. For example, the HE and DE scores were converted into percentages as total scores for these assessments tasks. Then, the scaled dataset was stored as a csv file for R processing to implement SVM-regression based algorithm on this pre-processed dataset. The SVM- regression modelling technique was used to develop the predictive model by using these three predictor variables. In addition, this study used K-fold cross-validation technique to assess the effect of predictor variables on predicted variable and to check if the model has been overfitted. Consequently, the mean squared error (MSE) and root-mean-square error (RMSE) metrics were used to weigh the differences between actual and predicted values by a model to measure the goodness of fit. It is identified that there is a gap between training and testing errors (Figure 2) and the average MSE-train error is lower than the average MSE-test error (Table 3). It means overfitting might have occurred. However, the difference between the RMSE of training and test data set errors of the model is not significant (Table 3). Figure 2 visualizes the supervised machine learning curves of training and test sets across all 5 trails of K-fold cross-validation.

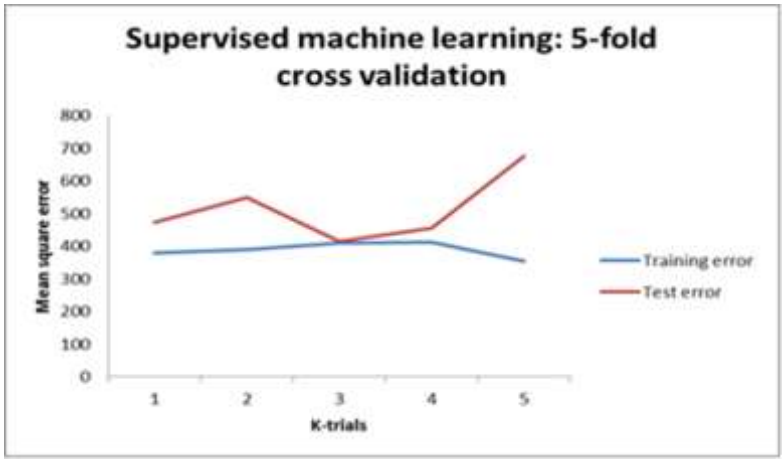

Figure 2: Recorded mean square errors across all k (k=5) trials of supervised machine learning

Table 3 presents the average MSE and RMSE values of both training and test sets recorded across all K trials to estimate how accurately a predictive model will perform in practice.

Table 3: Average MSE and RMSE results computed across 5-trails of cross-validation

Statistical measures	**Training set**	**Test set**
MSE (average of 5-trails)	185.8574	235.9989
RMSE (average of 5-trails)	13.62255	15.36225

In addition, the prediction accuracy (total number of accurate predictions) of the test data was calculated to find how well the model predicts the final exam scores of all students on new dataset. The total number of correct predictions in percentages (CPA) for the final programming exam was calculated by dividing the number of accurate predictions by total number of predictions. In this study, the accurate prediction (predicted value) is defined as; predicted exam score is within 90-110% of actual exam score.

Table 4 shows an example of identifying correction prediction extracted from trail 1 of test data set results of K-fold cross-validation. The total number of accurate predictions was 99 out 190 and the prediction accuracy of the model is 52%.

Table 4: Example of identifying accurate predictions – trail 1 of K-fold validation

K-trial	**Predicted score**	**Actual score**	**Difference in %**	**Accurate prediction (Yes / No)**
1	63	44	43.18	No
	55	53	3.77	Yes
	80	87	8.75	Yes
	45	34	32.35	No
	95	87	9.19	Yes
	64	70	10.0	No
..	..	..		..
Total number of correct predictions total of 5-trails)				99 out of 190 (52%)

Consequently, one of the objectives of this study is to predict students who are academically at-risk. That is, if the student gets final exam score below 50 out of 100, then he or she will be identified as academically at-risk. However, this study is aimed to identify students who are fail and marginal pass as well. Hence, this study tags students who secured less than or equal to 55 as at-risk students, to check the prediction accuracy of test set results. We tested the prediction accuracy of at-risk students based on test set of final scores, computed across 5-trials of cross-validation with actual final scores. Table 5 shows an example of identifying academically at-risk students extracted from trail 1 of test data set results of K-fold cross-validation.

Table 5: Example of identifying academically at-risk students – trail 1 of K-fold cross-validation

K-trial	Predicted score	Actual score	Is the student correctly identified as academically at-risk
1	63	44	No
	55	53	Yes
	91	47	No
	45	34	Yes
	83	57	No
	80	17	No
	34	55	Yes
..	..	..	..

Consequently, a binomial test was conducted to check whether the model's performance on predicting students at-risk is rational. The binomial test compares the number of correct predictions observed in a number of trials with a hypothesized probability of success (Stowell, 2012; Abdi, 2007). So, to test hypothesis developed for this study a binomial test was conducted based on number of successful predictions observed (on identifying at-risk students) over number of trails from test set results. The binomial test indicated that p value 0.00574 is < 0.050, which is significant, and the probability of success rate is 46% (0.462).

5. DISCUSSION

The main purpose of this study was to develop a SVM-regression model to predict students who might be at-risk of failure in programming courses. Our previous research studies confirmed that the predictor variables homework, demo exercises, and prior knowledge in programming were strongly related to programming exam performance (Veerasamy;D'Souza;Lindén;Kaila;Laakso;& Salakoski, 2016; Veerasamy, Daryl D'Souza, & Laakso, 2018). In addition, the multiple linear regression results (Table 2) of our study revealed that there is a significant relationship between theses predictor variables on the dependent variable, final exam. Therefore, we conclude that student prior programming knowledge, homework and demo scores together in a predictive model is a good fit of the data. Subsequently, this study attempted to explore the impact of formative assessment tasks and prior programming knowledge in predicting student's final exam scores. The results in Table 4 indicate that the success rate of the model is 52% on predicting student final exam scores of all students in the programming course. It shows that this model can predict students' final exam performance based on their prior knowledge levels and performance in formative assessment tasks such as homework and demo exercises, albeit with a prediction error of 48%. The statistical results of binomial test revealed that the model has a 46% success rate for predicting academically at-risk students. The comparison between MSE/RMSE values of training and validation sets (Table 3 and Figure 2) suggest that the model is slightly over fitted. However, the difference between training set and test set RMSE values is not significant. However, the mean square error of the training set is lower than the mean square error of the validation set, which should be further analyzed.

From these statistical results the following points emerged. First, our previous study findings (Veerasamy, Daryl D'Souza, & Laakso, 2018) and results of this study suggest that prior programming knowledge also has a marked effect on student learning outcomes, and can be included as one of the predictors of summative performance for programming courses. Second, in keeping with many prior studies (Veerasamy;D'Souza;Lindén;Kaila;Laakso;& Salakoski, 2016; Watson;Li;& Godwin, 2014; Huang, 2011), these multiple linear regression results confirmed for us formative assessment tasks can be considered as significant predictor variables for student academic performance. Third, although the average prediction accuracy of the model is not low, it seems including one or more necessary predictor variables in the model may improve its overall predictive accuracy. For example, problem-solving skills (PSS) and learning programming are interrelated and PSS is identified as a fundamental skill for computer science students (Lye & Koh, 2014; Uysal, 2014; Veerasamy;D'Souza;Lindén;& Laakso, 2018). So, PSS can be included as one of the predictors in the model to predict student performance. Forth, utilizing learning analytics (LA) and including student log data in the model may alleviate the accurate prediction of at-risk students in programming courses. For example, learning management system platforms eased the difficulty of capturing data for the events that occur in the learning environment and these learning management systems captured data set can be included in the models to predict students at-risk. Moreover, this kind of LA enhanced models can be deployed as an early warning system for educators to visualize student engagement and likelihood of success. However, this should be analyzed further.

Although the overall prediction accuracy of the model is good, the prediction accuracy results (52%) suggest that attention should be paid to the effects of the interaction between the selected variables, although SVM might have taken

care of it. For example, students who have no prior knowledge in programming did not perform well in formative assessment tasks may be considered at-risk. Furthermore, individual differences in prior knowledge in topic are important and those differences should be cross examined with other predictor variables of the study. However, formative assessment tasks may influence student achievement despite their prior programming knowledge. So, these points suggest us to look for multivariate regression techniques to clarify the role of the interaction effects in the prediction of final exam performance and or for final grades. Similarly, the percentage of accuracy predictions on identifying at-risk students is neither high nor low and that persuaded us to include two more research questions: (i) How does our model convincingly predict at-risk students? (ii) What factors might have impacted the prediction accuracy of our proposed model and how to develop a more accurate at-risk prediction model? First, the binomial test results confirm that this model can be considered as a non-parametric test by instructors to identify students who need assistance at early stages in the course. For example, if the goal of the instructor is to identify at-risk students based on their perceived prior knowledge levels and selected formative assessment scores, then it is possible by our proposed model as it might identify 46% of those at least. Moreover, our preliminary post analysis on these test results show that on average students who secured less than 70% in formative assessment scores with little or basic prior programming knowledge in programming may fail in the final programming exam and increase the prediction accuracy in identifying at-risk students from 46% to nearly 63%. So, we conjecture that analyzing student prior knowledge in topic and ongoing assessment task results would help instructors to identify students that need assistance in the course.

On the other hand, our predication accuracy results raised another question. There have been studies that listed the common factors that impact the prediction accuracy of the statistical models (O.Ogundimu;G.Altman;& S.Collins, 2016; Kattan, 2011). First, sample size; there are no strict rules for minimum sample size requirements for predictive models. However, the number of samples and number of attributes may influence model prediction accuracy in regression analysis (Peduzi, Concato, Kemper, Holford, & Feinstein, 1996). However, as noted, SVM performs well on small datasets. Furthermore, the dataset of this study was collected systematically via ViLLE and preprocessed to assure that collected data is complete, consistent and does not contain any errors. We, therefore, surmise that our study sample is complete, although we need to reanalyze the structure of it. Second, selection of predictors; selection of variables often depends on research area and goals. Therefore, it is important to decide what and how many variables must be included for predictive models. Notably, even unnecessary predictors may raise prediction errors in the model. Similarly, the use of too many variables that provide similar information will bring the issue of multicollinearity and certainly affect the model's goodness of fit. Hence, researchers often use regression analysis to select important variables for predictive analysis (Derksen & Keselman, 1992). So, to keep these things in perspective, this study selected predictors based on past research studies, learning theories, and multiple regressions – stepwise procedures. In addition, we conducted the Variance inflation factor (VIF) statistical analysis to check whether our model has any highly correlated predictor variables that might cause multicollinearity. The VIF for the predictor variables PPK, HE and DE all are in between 1.00 and 1.4, and these results suggest that our developed model has no multicollinearity issues. However, it seems that student's lack of "learning transferability" skills might have impacted their final programing exam performance despite the good scores received/secured in continues assessments and has impacted the prediction accuracy of the model. However, it should be analyzed further. Third, model fitting; it is obviously important for researchers to ensure that developed models fit well or can be generalized. That is, the model fit to the training set with significant prediction accuracy is also expected to make reliable predictions on unknown or test data. However, if the model is overfitted or under fitted then such a model would have poor prediction accuracy with high prediction error. Our K-fold cross-validation results suggest that the "model is slightly overfitted". It could have occurred due to the sample size used in this study which should be analyzed further. Although our model's overall prediction accuracy is good, it is necessary to enhance the model based on the details given above to improve the prediction accuracy. That is, first it is required to restructure the model approach to boost the accuracy of a model. Second, insufficient sample size, model overfitting, and the lack of predictors could have impacted the prediction accuracy, which should be analyzed further. For example, adding more data, treating missing and outlier values in the training dataset (require deployment of analytical measures), tuning parameters and application of bootstrap techniques may increase the prediction accuracy of the model. Third, this study used K-fold cross-validation, which allows for more accurate estimates and guarantees testing on every single data instance which makes it computationally intensive (Witten & Frank, 2005). Hence, the validation across the data set would have affected the accuracy found for the model we developed. Furthermore, K-fold technique has the tendency to overestimate the extra sample error unlike Leave-one-out cross-validation technique (Borra & Ciaccio, 2010) that could have turned our predictive model marginally overfitted. Thus, we surmise that, as K-fold cross validation has potential conservative bias, it would have impacted the prediction accuracy of the model. However, this should be analyzed further. Consequently, we also conclude that deployment of Corrected-K-fold cross-validation may reduce the bias of the estimation and the possibility to increase our model's performance (Asif, Merceron, & Pathan, 2015; Huang & Fang, 2013; Borra & Ciaccio, 2010). Finally, SVM-regression model of this study yielded considerable results on predicting student final programming exam performance, though these results pushed us to get more optimal tuning parameters to improve the model performance.

6. CONCLUSION AND LIMITATIONS OF THE STUDY

Our research studies identified that formative assessment tasks and prior programming knowledge play significant roles in predicting student final programming exam performance. Notably, our SVM, a data mining technique based predictive model yielded sizeable accurate prediction accuracy to predict student academic performance. However, the prediction success rate of identifying at-risk students is not significant. On the other hand, our post preliminary analysis on test results suggests that student who secured less than 70% in assessment tasks with little or basic prior knowledge in programming can be assumed as at-risk students and increase the prediction accuracy in identifying at-risk students from 46% to nearly 63%. Hence, these results provide immediate information for programming course instructors to facilitate the learning for students at-risk. In addition, these results may help instructors to develop interventions to improve success of students. For example, instructors deliver decisions / suggestions to at-risk students based on the predicted results. Finally data and the results of this study will be used for our further research to build an accurate predictive model for student academic performance in programming courses.

Like all research, this study has several limitations and it is not free from its weaknesses. First, the subjects of study were 190 introductory programming students of our university. Second, although the average prediction accuracy and confidence interval of the model is adequate, the model is slightly overfitted. Third, there is a possibility that the statistical significance of the identified predictor variables of a model may be affected by external factors. For example, lack of interest, self-confidence, family/peer pressure and etc. Consequently, it is hard to generalize the predictive model of this study to all programming students even though our SVM-regression based predictive model generated considerable results. However, this model can be applied to other courses those has continuous summative tasks and final exam as assessment components to predict student performance. The predictive model introduced in this research paper provides a natural guide to future work. For example, three major issues that were addressed in this research can be taken into account to conduct another research in future. First, as noted, including some other academic variables that impact student performance in programming courses. For example, previous study success or grade point average earned by student in other courses was widely used as one the significant predictor variables in many predictive models to measure student academic performance (Huang & Fang, 2013). So, the possible research question is: Does student previous study success impact student performance in programming courses? Second, programming requires students to have a good understanding of programming concepts and meta-cognitive skills in order to be proficient in programming (Uysal, 2014). Hence, in addition to prior programming knowledge, including course specific factors such as, student problem solving skills, computational thinking levels, programming aptitude test scores, and student performance on programming topics that identified by the Delphi concept inventory as predictor variables may minimize the bias and variance of the predictive model. Thus, the research model may be aligned, by adding these variables to determine if there is any improvement in prediction accuracy towards identifying at-risk students at the early stages of programming course. Third, as we suggested, deployment of appropriate cross validation techniques with significant factors may provide a more accurate representation of the model's predictive performance.

7. EDUCATIONAL IMPLICATIONS

Despite these limitations, our findings provide some suggestions for learners, instructors, course administrators, and academic advisors. Prior knowledge in topic strongly related to student performance and implies that it has substantial impact on the learning process. So, it is important to assess student prior knowledge and such information can be used by instructors to facilitate the learning situation, student engagement, and critical thinking in the course. In addition, knowing student prior knowledge levels early in the semester might help academic advisors to observe and intervene with students. Similarly, formative assessment tasks play vital role in student learning outcomes. Therefore, gauging student prior knowledge and assessment results, instructors can monitor and evaluate student learning outcomes. Furthermore, by obtaining these results early in the course, instructors can identify students who need attention to focus on related supporting activities. As noted, educational institutions often use predictive models as early warning systems to identify low motivated learners who need more attention than others in order to reduce the drop-out rates. However, Internet and educational technologies have changed the concepts of educational assessment practices and forced educators to adopt new approaches to develop predictive models that could help them to improve active learning and communication between students to alleviate their learning issues. As such, this study affirms that the data extracted via learning tools can be used in order to build models student retention/performance.

8. ACKNOWLEDGEMENT

The authors wish to thank all members of ViLLE team research project group, Department of Future Technologies, University of Turku for their comments and support that greatly improved the manuscript. In addition, authors would like to show their gratitude to Dr. Tapio Pahikkala, Assistant Professor at Department of Future Technologies who participated in the research discussion of this study (SVM-regression algorithm) to improve the manuscript. This work was supported by the ViLLE Team Research Project and by the University of Turku, Turku, Finland.

9. REFERENCES

[1] Abdi, H., 2007. *Binomial distribution: binomial and sign tests.* s.l.:Encyclopedia of Measurement and Statistics.

[2] Abu-Oda, G. S. & El-Halees, A. M., 2015. Data Mining in Higher Education: University student dropout case study. *International Journal of Data Mining & Knowledge Management process,* 5(1), pp. 15-27.

[3] Alexandron, G., Armoni, M., Gordon, M. & Harel, D., 2012. *The effect of Previous Programming Experience on the Learning of Scenario-Based Programming.* s.l., ACM, pp. 151-159.

[4] Ali, A. & Smith, D., 2014. Teaching an Introductory Programming Language. *Journal of Information Technology Education: Innovations in Practice,* Volume 13, pp. 57-67.

[5] Asif, R., Merceron, A. & Pathan, M. K., 2015. Predicting Student Academic Performance at Degree Level: A Case Study. *International Journal of Intelligent Systems and Applications,* 7(1), pp. 49-61.

[6] Astin, A. W., 1978. *Four Critical Years. Effects of College on Beliefs, Attitudes, and Knowledge.* s.l.:ERIC.

[7] Ausubel, D. P., Novak, J. D. & Hanesian, H., 1978. *Educational Psychology: A cognitive view.* New York: Rinehart and Winston.

[8] Bergin, S., Mooney, A., Ghent, J. & Quille, K., 2015. Using Machine Learning Techniques to Predict Introductory Programming Performance. *International Journal of Computer Science and Software Engineering,* December, 4(12), pp. 323-328.

[9] Bergin, S. & Reilly, R., 2005. *Programming: factors that influence success.* s.l., s.n.

[10] Borra, S. & Ciaccio, A. D., 2010. Measuring the prediction error. A comparison of cross-validation, bootstrap and covariance penalty methods. *Computational Statistics and Data Analysis,* Volume 54, pp. 2976-2989.

[11] Byrne, P. & Lyons, G., 2001. *The effect of student attributes on success in programming.* Canterbury, UK, ACM, pp. 49-52.

[12] Conjin, R., Snijders, C. & Kleingeld, A., 2017. Predicting Student Performance from LMS Data: A Comparison of 17 Blended Courses Using Moodle LMS. *IEEE Transactions on Learning Technologies* , January_March, 10(1), pp. 17-29.

[13] Corbett, A. T. & Anderson, J. R., 2001. *Locus of feedback control in computer-based tutoring: Impact on learning rate, achievement and attitudes.* New York, ACM, pp. 245-252.

[14] Costa, E. B., Fonseca, B., Santana, M. A. & de, F. F., 2017. Evaluating the effectiveness of educational data mining techniques for early prediction of students' academic failure in introductory programming courses. *Computers in Human Behavior,* Volume 73, p. 247–256.

[15] de-la-Fuente-Valentín, L., Pardo, A. & Kloos, C. D., 2013. Addressing drop-out and sustained effort issues with large practical groups using an automated delivery and assessment system. *Computers & Education,* 61(February), pp. 33-42.

[16] Derksen, S. & Keselman, H., 1992. Backward, forward and stepwise automated subset selection algorithms: Frequency of obtaining authentic and noise variables. *British Journal of Mathematical and Statistical Psychology,* 45(2), pp. 265-282.

[17] Devasia, T., P, V. T. & Hegde, V., 2016. *Prediction of students performance using Educational Data Mining.* Ernakulam, IEEE, pp. 91-95.

[18] Dickey, D. A., U., N. C. S. & Raleigh, N., 2012. *Introduction to Predictive Modeling with Examples.* s.l.:s.n.

[19] EIGamal, A., 2013. An Educational Data Mining Model for Predicting Student Performance in Programming Course. *International Journal of Computer Applications,* May, 70(17), pp. 22-28.

[20] Evans, G. E. & Simkin, M. G., 1989. What Best Predicts Computer Proficiency?. *Communications of the ACM,* 32(11), pp. 1322-1327.

[21] Fortmann-Roe, S., 2012. *Accurately measuring model prediction error.* s.l.:s.n.

[22] Grover, S., Pea, R. & Cooper, S., 2016. *Factors Influencing Computer Science Learning in Middle School.* Memphis, TN, USA, ACM, pp. 552-557.

[23] Guo, B. et al., 2015. *Predicting Students Performance in Educational Data Mining.* Wuhan, China, s.n.

[24] Hailikari, T., 2009. *Assessing university students' prior knowledge implications for theory and practice,* Helsinki: Helsinki University Print, Finland.

[25] Holden, E. & Weeden, E., 2003. *The impact of prior experience in an information technology programming course sequence.* Lafayette, Indiana, ACM, pp. 41-46.

[26] Hsu, W. C. & Plunkett, S. W., 2016. *Attendance and Grades in Learning Programming Classes.* Canberra, s.n.

[27] Huang, S., 2011. *Predictive Modeling and analysis of Student Academic Performance in an Engineering Dynamics Course,* Logan, Utah: Utah State University.

[28] Huang, S. & Fang, N., 2013. Predicting student academic performance in an engineering dynamics course: A comparison of four types of predictive mathematical models. *Computers & Education,* 61(1), p. 133–145.

[29] Hämäläinen, W. & Vinni, M., 2006. *Comparison of Machine Learning Methods for Intelligent Tutoring Systems.* Jhongli, Taiwan, Springer, pp. 525-534.

[30] Jacoby, J. & Matell, M. S., 1971. Three-point Likert Scales Are Good Enough. *Journal of Marketing Research,* 8(4), pp. 495-500.

[31] Kattan, M. W., 2011. Factors affecting the Accuracy of Prediction Models Limit the Comparison of Rival Prediction Models When Applied to Separate Data Sets. *European Urology,* 59(4), pp. 566-567.

[32] Kebritchi, M., Hirumi, A. & Bai, H., 2010. The effects of modern mathematics computer games on mathematics achievement and class motivation. *Computers & Education,* 55(2), pp. 427-443.

[33] Kim, J.-H., 2009. Estimating classification error rate: Repeated cross-validation, repeated hold-out and bootstrap. *Computational Statistics and Data Analysis,* Volume 53, pp. 3735-3745.

[34] Kinnunen, P. & McCartney, R., 2007. *Through the eyes of instructors: a phenomenographic investigation of student success.* Atlanta, ACM, pp. 61-72.

[35] Koulouri, T., Lauria, S. & Macredie, R. D., 2014. Teaching Introductory Programming: A Quantitative Evaluation of Different Approaches. *ACM Transactions on Computing Education (TOCE),* 14(4), pp. 26.1-26.27.

[36] Krumm, A. E., Waddington, R. J., Teasley, S. D. & Lonn, S., 2014. A learning Management System-Based Early Warning System for Academic Advising in Undergraduate Engineering. In: J. A. Larusson & B. White, eds. *Learning Analytics: From Research to Practice.* New York: Springer, pp. 103-119.

[37] Lee, Y.-J., 2016. Predicting Students' Problem Solving Performance using Support Vector Machine. *Journal of Data Science,* 14(2), pp. 231-244.

[38] Lin, T.-F. & Chen, J., 2006. Cumulative class attendance and exam performance. *Applied Economics Letters,* 13(14), pp. 937-942.

[39] Longi, K., 2016. *Exploring factors that affect performance on introductory programming courses,* Helsinki: s.n.

[40] Lye, S. Y. & Koh, J. H. L., 2014. Review on teaching and learning of computational thinking through programming: What is next for K-12?. *Computers in Human Behavior,* Volume 41, pp. 51-6141.

[41] Marbouti, F., Diefes-Dux, H. A. & Madhavan, K., 2016. Models for early prediction of at-risk students in a course using standards-based grading. *Computers & Education,* Volume 103, pp. 1-15.

[42] Marios Tacio Silva, E. d. B. C. E. T. S. P. H. B. J. C., 2014. *Failure rates in introductory programming: A 2006–2012 study at a Brazilian University.* Madrid, Spain, IEEE, pp. 1-7.

[43] O.Ogundimu, E., G.Altman, D. & S.Collins, G., 2016. Adequate sample size for developing prediction models is not simply related to events per variable. *Journal of Clinical Epidemiology,* Volume 76, pp. 175-182.

[44] Pardo, A., 2014. Designing Learning Analytics Experiences. In: J. Larusson & B. White, eds. *Learning Analytics From Research to Practice.* New York: Springer, pp. 15-35.

[45] Peduzi, P. et al., 1996. A Simulation Study of the Number of Events per Variable in Logistic Regression Analysis. *Journal of Clinical Epidemiology,* 49(12), pp. 1373-1379.

[46] Rajala, T. & Erkki Kaila, M.-J. L., 2005. *ViLLE.* [Online] Available at: https://ville.utu.fi/ [Accessed 20 10 2015].

[47] Romero, C., López, M.-I., Luna, J.-M. & Ventura, S., 2013. Predicting students' final performance from participation in on-line discussion forums. *Computers & Education,* Volume 68, p. 458–472.

[48] Rosenschein, J. S., Vilner, T. & Zur, E., 2004. *Work in progress: programming knowledge - does it affect success in the course introduction to computer science using Java.* Savannah, GA, IEEE Xplore, pp. 3-4.

[49] Seery, M. K., 2009. *The Effect of Prior Knowledge in Undergraduate Performance in Chemistry: A Correlation–Prediction Study,* Dublin: Dublin Institute of Technology.

[50] Stowell, S., 212. *Performing a binomial test in R.* s.l.:s.n.

[51] Su, A. Y. S. et al., 2015. Effects of Annotations and Homework on Learning Achievement: An Empirical Study of Scratch Programming Pedagogy. *Journal of Educational Technology & Society,* 2015 October, 18(4), pp. 331-343.

[52] Tafliovich, A., Campbell, J. & Petersen, A., 2013. *A Student Perspective on Prior Experience in CS1*. Denver, Colorado, USA., ACM, pp. 239-244.

[53] Uysal, M. P., 2014. Improving First Computer Programming Experiences: The Case of Adapting a Web-Supported and Well- Structured problem-Solving Method to a Traditional Course. *Contemporary Educational Technology,* 5(3), pp. 198-217.

[54] Vapnick, V. N., 1995. *Statistical Learning Theory.* London: A Wiley-Interscience.

[55] Watson, C., Li, F. W. & Godwin, J. L., 2014. *No tests required: comparing traditional and dynamic predictors of programming success.* s.l., ACM, pp. 469-474.

[56] Veerasamy, A. K., Daryl D'Souza, R. L. & Laakso, M.-J., 2018. The impact of prior programming knowledge on lecture attendance and final exam. *Journal of Educational Computing Research,* 0(0), pp. 226-253.

[57] Veerasamy, A. K. et al., 2016. The Impact of Lecture Attendance on Exams for Novice Programming Students. *International Journal of Modern Education and Computer Science (IJMECS),* 8(5), pp. 1-11.

[58] Veerasamy, A. K., D'Souza, D., Lindén, R., & Laakso, M.-J. (2018, November 6). Relationship between perceived problem-solving skills and academic performance of novice learners in introductory programming courses. *Journal of Computer Assisted Learning*.

[59] Vihavainen, A., 2013. *Predicting Students' Performance in an Introductory Programming Course Using Data from Students' Own Programming Process.* Beijing, China, IEEE, pp. 498-499.

[60] Witten, I. H. & Frank, E., 2005. Credibility: Evaluating what's been learned. In: J. Gray, ed. *Data Mining - Practical Machine learning tools and techniques.* s.l.:Morgan Kaufmann, pp. 149-151.

[61] Vogel-Heuser, B., Rehberger, S., Frank, T. & Aicher, T., 2014. *Quality despite quantity — Teaching large heterogenous classes in C programming and fundamentals in computer science.* Istanbul, IEEE.

[62] Wong, W.-c., 2014. *The Impact of Programming Experience on Successfully Learning Systems Analysis and Design.* Baltimore, Maryland USA, Education Special Interest Group of AITP, pp. 1-9.

[63] Yuer, A. & Güngörmüş, A. H., 2011. Factors Associated with Student Performance in Financial Accounting Course. *European Journal of Economic and Political Studies,* 4(2), pp. 141-156.

[64] Zingaro, D., 2015. Examining Interest and Grades in Computer Science 1: A Study of Pedagogy and Achievement Goals. *ACM Transactions on Computing Education,* September, 15(3), pp. 14:01 -14:18.

[65] Öncu, S., Sengel, E. & Delialioglu, Ö., 2008. *How does prior knowledge affect student engagement in undergraduate level computer literacy classes?*. Eskişehir, Turkey, IETC, pp. 1063-1067.