TUCS

Ashok Kumar Veerasamy

Predictive Models As Early Warning Systems For Student Academic Performance In Introductory Programming

Turku Centre for Computer Science

TUCS Dissertations
No 259, December 2020

Predictive models as early warning systems for student academic performance in introductory programming

Ashok Kumar Veerasamy

To be presented, with the permission of the Faculty of Science and Engineering, for public criticism in the Auditorium XX1 (Agora) on December 8th, 2020 at 12:00.

University of Turku
Department of Future Technologies
Vesilinnantie 5, 20500 Turku, Finland

2020

Supervised by

Professor Tapio Salakoski
Department of Future Technologies
University of Turku
Turku, Finland

Associate Professor Mikko-Jussi Laakso
Department of Future Technologies
University of Turku
Turku, Finland

Senior Lecturer Daryl D'Souza
School of Science & Information Technology
RMIT University
Melbourne, Australia

Reviewed by

Senior University Lecturer Jaakko Hollmén
Department of Computer Science
Aalto University
Helsinki, Finland

Senior Lecturer Matthew Butler
Department of Human Centred Computing
Monash University
Melbourne, Australia

Opponent

Associate Professor Petri Ihantola
Department of Education
University of Helsinki
Helsinki, Finland

The originality of this thesis has been checked in accordance with the University of Turku quality assurance system using the **Turnitin** Originality Check service.

ISBN 978-952-12-4014-0
ISSN 1239-1883

Abstract

Computer programming is fundamental to Computer Science and IT curricula. At the novice level it covers programming concepts that are essential for subsequent advanced programming courses. However, introductory programming courses are among the most challenging courses for novices and high failure and attrition rates continue even as computer science education has seen improvements in pedagogy. Consequently, the quest to identify factors that affect student learning and academic performance in introductory computer programming courses has been a long-standing activity. Specifically, weak novice learners of programming need to be identified and assisted early in the semester in order to alleviate any potential risk of failing or withdrawing from their course. Hence, it is essential to identify at-risk programming students early, in order to plan (early) interventions.

The goal of this thesis was to develop a validated, predictive model(s) with suitable predictors of student academic performance in introductory programming courses. The proposed model utilises the Naïve Bayes classification machine learning algorithm to analyse student performance data, based on the principle of parsimony. Furthermore, an additional objective was to propose this validated predictive model as an early warning system (EWS), to predict at-risk students early in the semester and, in turn, to potentially inform instructors (and students) for early interventions.

We obtained data from two introductory programming courses in our study to develop and test the predictive models. The models were built with student presage and in progress-data for which instructors may easily collect or access despite the nature of pedagogy of educational settings. In addition, our work analysed the predictability of selected data sources and looked for the combination of predictors, which yields the highest prediction accuracy to predict student academic performance. The prediction accuracies of the models were computed by using confusion matrix data including overall model prediction accuracy, prediction accuracy sensitivity and specificity, balanced accuracy and the area under the ROC curve (AUC) score for generalisation.

On average, the models developed with formative assessment tasks, which were partially assisted by the instructor in the classroom, returned higher at-risk prediction accuracies than the models developed with take-home assessment task only as predictors. The unknown data test results of this study showed that it is possible to predict 83% of students that need support as early as Week 3 in a 12-week introductory programming course. The ensemble method-based results suggest that it is possible to improve overall at-risk prediction performance with low false positives and to incorporate this in early warning systems to identify students that need support, in order to provide early intervention before they reach critical stages (at-risk of failing).

The proposed model(s) of this study were developed on the basis of the principle of parsimony as well as previous research findings, which accounted for variations in academic settings, such as academic environment, and student demography. The predictive model could potentially provide early warning indicators to facilitate early warning intervention strategies for at-risk students in programming that allow for early interventions. The main contribution of this thesis is a model that may be applied to other programming and non-programming courses, which have both continuous formative and a final exam summative assessment, to predict final student performance early in the semester.

Tiivistelmä

Ohjelmointi on informaatioteknologian ja tietojenkäsittelytieteen opinto-ohjelmien olennainen osa. Aloittelijatasolla opetus kattaa jatkokurssien kannalta keskeisiä ohjelmoinnin käsitteitä. Tästä huolimatta ohjelmoinnin peruskurssit ovat eräitä haasteellisimmista kursseista aloittelijoille. Korkea keskeyttämisprosentti ja opiskelijoiden asteittainen pois jättäytyminen ovat vieläkin tunnusomaisia piirteitä näille kursseille, vaikka ohjelmoinnin opetuksen pedagogiikka onkin kehittynyt. Näin ollen vaikuttavia syitä opiskelijoiden heikkoon suoriutumiseen on etsitty jo pitkään. Erityisesti heikot, aloittelevat ohjelmoijat tulisi tunnistaa mahdollisimman pian, jotta heille voitaisiin tarjota tukea ja pienentää opiskelijan riskiä epäonnistua kurssin läpäimisessä ja riskiä jättää kurssi kesken. Heikkojen opiskelijoiden tunnistaminen on tärkeää, jotta voidaan suunnitella aikainen väliintulo.

Tämän väitöskirjatyön tarkoituksena oli kehittää todennettu, ennustava malli tai malleja sopivilla ennnustusfunktioilla koskien opiskelijan akateemista suoriutumista ohjelmoinnin peruskursseilla. Kehitetty malli käyttää koneoppivaa naiivia bayesilaista luokittelualgoritmia analysoimaan opiskelijoiden suoriutumisesta kertynyttä aineistoa. Lähestymistapa perustuu yksinkertaisimpien mahdollisten selittävien mallien periaatteeseen. Lisäksi, tavoitteena oli ehdottaa tätä validoitua ennustavaa mallia varhaiseksi varoitusjärjestelmäksi, jolla ennustetaan putoamisvaarassa olevat opiskelijat opintojakson alkuvaiheessa sekä informoidaan ohjaajia (ja opiskelijaa) aikaisen väliintulon tarpeellisuudesta.

Keräsimme aineistoa kahdelta ohjelmoinnin peruskurssilta, jonka pohjalta ennustavaa mallia kehitettiin ja testattiin. Mallit on rakennettu opiskelijoiden ennakkotietojen ja kurssin kestäessä kerättyjen suoriutumistietojen perusteella, joita ohjaajat voivat helposti kerätä tai joihin he voivat päästä käsiksi oppilaitoksesta tai muusta ympäristöstä huolimatta. Lisäksi väitöskirjatyö analysoi valittujen datalähteiden ennustettavuutta ja sitä, mitkä mallien muuttujista ja niiden kombinaatioista tuottivat kannaltamme korkeimman ennustetarkkuuden opiskelijoiden akateemisessa suoriutumisessa. Mallien ennustusten tarkkuuksia laskettiin käyttämällä sekaannusmatriisia, josta saadaan laskettua ennusteen tarkkuus, ennusteen spesifisyys, sensitiivisyys, tasapainotettu tarkkuus sekä luokitteluvastekäyriä (receiver operating characteristics (ROC)) ja näiden luokitteluvastepinta-ala (area under curve (AUC))

Mallit, jotka kehitettiin formatiivisilla tehtävillä, ja joissa ohjaaja saattoi osittain auttaa luokkahuonetilanteessa, antoivat keskimäärin tarkemman ennustuksen putoamisvaarassa olevista opiskelijoista kuin mallit, joissa käytettiin kotiin vietäviä tehtäviä ainoina ennusteina. Tuntemattomalla testiaineistolla tehdyt

mallinnukset osoittavat, että voimme tunnistaa jo 3. viikon kohdalla 83% niistä opiskelijoista, jotka tarvitsevat lisätukea 12 viikkoa kestävällä ohjelmoinnin kurssilla. Tulosten perusteella vaikuttaisi, että yhdistämällä metodeja voidaan saavuttaa parempi yleinen ennustettavuus putoamisvaarassa olevien opiskelijoiden suhteen pienemmällä määrällä väärin luokiteltuja epätositapauksia. Tulokset viittaavat myös siihen, että on mahdollista sisällyttää yhdistelmämalli varoitusjärjestelmiin, jotta voidaan tunnistaa avuntarpeessa olevia opiskelijoita ja tarjota täten varhaisessa vaiheessa tukea ennen kuin on liian myöhäistä.

Tässä tutkimuksessa esitellyt mallit on kehitetty nojautuen yksinkertaisimman selittävän mallin periaatteeseen ja myös aiempiin tutkimustuloksiin, joissa huomioidaan erilaiset akateemiset ympäristöt ja opiskelijoiden tausta. Ennustava malli voi tarjota indikaattoreita, jotka voivat mahdollisesti toimia pohjana väliintulostrategioihin kurssilta putoamisvaarassa olevien opiskelijoiden tukemiseksi. Tämän tutkimuksen keskeisin anti on malli, jolla opiskelijoiden suoriutumista voidaan arvioida muilla ohjelmointia ja muita aihepiirejä käsittelevillä kursseilla, jotka sisältävät sekä jatkuvaa arviointia että loppukokeen. Malli ennustaisi näillä kursseilla lopullisen opiskelijan suoritustason opetusjakson alkuvaiheessa.

Acknowledgements

I am deeply indebted to my supervisors Prof. Tapio Salakoski, Associate Prof. Mikko-Jussi Laakso and Dr. Daryl D'Souza for supporting me throughout my research career with their patience, guidance and immense knowledge. Moreover, all of my supervisors' flexible approach eased me to express and present my research findings with clarity and to complete writing this thesis in a reasonable time frame. I also would like to thank thesis reviewers Dr. Matthew Butler and Dr. Jaakko Hollmén for their valuable comments and efforts towards improving my thesis.

I would like to express my special thanks and appreciation to Mr. Rolf Lindén, a fellow ex-research scholar for assisting me with statistical analysis for this research. I am also deeply grateful to Prof. Tapio Pahikkala and Dr. Paavo Nevalainen for sparing their valuable time whenever I approached them and showing me the way ahead.

I am grateful to my colleagues Mr. Erno Lokkila, Mr. Henri Kajasilta, Mr. Peter Larsson, Mr.Einari Kurvinen, Mr. Teemu Rajala, Ms. Petra Enges-Pyykönen, Dr. Erkki Kaila, and Dr. Mikko-Ville Apiola for their support throughout the course of this research. I also wish to thank all members of my lovely ViLLE team for their tremendous support that greatly helped me to complete this study.

I would like to thank my special friends and ex-colleagues Dr. Anthony de Souza-Daw - Melbourne Polytechnic, Dr. Rajeev Kumar Kanth - Savonia University of Applied Sciences, Mr. Nanda Kumar - University of Turku, Dr. Tri Lam - University of Queensland, Mr. Anandavel Kannan – UN-Mission, Ms. Anuradha Arun Narayan, Mr. Vinayaga Murthy, and Mr & Mrs. Asanka Wanninayake for your support, kindness, and comforting throughout the ups and downs of my study, career and life.

My acknowledgement would be incomplete without thanking the biggest source of my strength, my lovely kids Miss. Kanishka Ranjani and Mr. Lingesh Kumar. Most of all, I dedicate this work to the soul of my father who taught me the meaning of life, my mother who taught me to be an independent and determined person, and to siblings who always love and care for me.

Contents

List of abbreviations

ALG	Algorithms and Programming
ATSE	At-risk prediction sensitivity
ATSP	At-risk prediction specificity
AUC	The area under the ROC curve
BAC	Balanced accuracy
DE	Demo exercises
EDM	Educational data mining
EWS	Early warning system
FE	Final exam
FEG	Final exam grade
FN	False negative
FP	False positive
HE	Homework exercises
INT	Introduction to Programming
KNN	K-Nearest neighbor
LA	Learning analytics
LEA	Lecture attendance
LMS	Learning management system
MAC	Model's overall prediction accuracy
ML	Machine learning
NBC	Naive Bayes classification
NPV	Negative predictive value
PSS	Problem-solving skills
PPK	Prior programming knowledge
PPV	Positive predictive value
ROC	Receiver operating characteristic
RQ	Research question
TN	True negative
TP	True positive
TT	Tutorial exercises

Nomenclature

PSI	Problem solving inventory / Problem solving skills

List of original publications

P1. Veerasamy, A. K., D'Souza, D., Lindén, R., Kaila, E., Laakso, M.-J., & Salakoski, T. (2016). The Impact of Lecture Attendance on Exams for Novice Programming Students. *International Journal of Modern Education and Computer Science (IJMECS), 8*(5), 1-11. doi:10.5815/ijmecs.2016.05.01

P2. 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, 56*(2), 226-253. doi:10.1177/0735633117707695

P3. 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, 35*(2), 246-255. doi:10.1111/jcal.12326

P4. Veerasamy, A. K., D'Souza, D., Lindén, R., & Laakso, M.-J. (2019, February 1). Prediction of Student Final Exam Performance in an Introductory Programming Course: Development and Validation of the Use of a Support Vector Machine-Regression Model. *Asian Journal of Education and E-learning, 7*(1), 1-14.

P5. Veerasamy, A. K., D'Souza, D., Lindén, R., & Laakso, M.-J. & Salakoski, T. (2019) Predictive Models as Early Warning Systems: A Bayesian classification model to identify at-risk students of programming. Computing Conference-2021 *(Accepted)*.

Other publications not included in this thesis

O1. Veerasamy, A.K., D'Souza D., Apiola M-V., Laakso M.-J., and Salakoski T. (2020) Using early assessment performance as early warning signs to identify at-risk students in programming courses – full paper was accepted and presented at 23.10.2020 in IEEE-FIE 2020.

O2. Veerasamy, A. K., D. D., & Laakso, M.-J. (2016). Identifying Novice Student Programming Misconceptions and Errors From Summative Assessments. *Journal of Educational Technology Systems, 45*(1), 50-73. doi:10.1177/0047239515627263.

O3. Veerasamy, A. K., & Shillabeer, A. (2014). Teaching English Based Programming Courses to English Learners/Non-Native Speakers of English. *ICEMI 2014 : 2014 3rd International Conference on Education and Management. 70*, pp. 17-22. Hong Kong: International Proceedings of Economics Development and Research. doi: 10.7763/IPEDR. 2014. V70. 4

O4. Veerasamy, A. K., & Souza-Daw, T. d. (2012). Impact of ICT on Society - Higher Education students in South-East Asia. *IEEE Symposium on Business, Engineering and Industrial Applications* (pp. 275-278). Bandung: IEEE.

O5. Veerasamy, A. K., (2008). Information System Research and Education in Developing Countries: Papua New Guinea A case study. *Pre- ICIS conference.* Paris. Retrieved from http://www-public.imtbs-tsp.eu/~assar/pre-ICIS08/venue.html

Chapter 1

Introduction

This chapter provides the background of introductory programming education and the motivation for developing machine learning models to identify at-risk students. Studies in teaching and learning programming, identification of factors that influence student success in programming, research questions, scope and significance of the thesis, are presented.

1.1 Introductory programming education

Programming is fundamental to computer science and cognate disciplines and is typically offered as a major to students of other disciplines. It is an essential basis for many other advanced computer science and engineering courses that follows in the subsequent years. Introductory programming courses are taught essentially in all universities to introduce principles of computer science and begin to develop programming skills.

However, the question "how to code a program in a computer language?" presents various challenges and difficulties to students and instructors. Programming has been identified as difficult to learn by novice students, and remains challenging, despite improvements in pedagogy, and ably supported by new technologies. Specifically, much research into improving teaching and learning of introductory programming has taken place (Luxton-Reilly, et al., 2018). Failure and attrition rates in programming continue to be in between 28-32% worldwide (Watson & Li, 2014; Bennedsen & Caspersen, 2019). A number of studies have been carried out to determine the factors that influence academic performance in programming courses, to establish why learning to program easier for some, more so than for others (Longi, 2016; Idemudia;Dasuki;& Ogedebe, 2016). In addition, several studies have attempted to construct effective models to predict student performance in programming courses to facilitate better interventions (Ahadi, Lister, Haapala, & Vihavainen, 2015; Carter, Hundhausen, & Adesope, 2015; Castro-Wunsch;Ahadi;& Petersen, 2017; Conijn;Snijders;& Kleingeld, 2017; Liao;Zingaro;Alvarado;Griswold;& Porter, 2019). However, the predictor variables used in these various models, and the models themselves, varies from one context to another, with variations occurring in student cohort, cultural setting, class size and classroom and academic environments. It is widely accepted that parsimony is important in model building (Vandekerckhove;Matzke;& Wagenmakers, 2014). However, these studies did not use parsimonious models to characterize or model the data with a minimum number of predictor variables. Moreover, many studies are in need of further verification due to inconsistencies in results obtained over a range of identified factors and educational data mining techniques (Costa, Fonseca, Santana, & Araújo, 2017). Therefore, computer science educators are often searching for key factors that can serve as performance indicators or predictor variables to identify dropout/at-risk students. Moreover, identifying student at risk of disengaging early in the semester would help instructors to execute timely interventions.

Consequently, one of the goals of this study was to identify potential predictors whilst maintaining a balance between parsimony and goodness of the model fit. In addition, identifying and choosing suitable machine learning techniques is also vital in developing predictive models. This is because machine reasoning allows a system to make inference based decisions about data. Moreover, machine learning is concerned with developing methods to discover models or patterns of data, which is significantly helpful in decision-making. The application of machine learning techniques in predicting student performance proved to be useful for identifying at-risk students and enable instructors to draw sense making decisions (Quille & Bergin, 2018; Liao;Zingaro;Alvarado;Griswold;& Porter, 2019; Liao, et al., 2019).

This thesis presents a study that focuses on developing, validating, and testing the Naïve Bayes classification (NBC) algorithm based predictive models, which may be employed to predict student performance and to identify at-risk students in introductory programming. NBC is a simple supervised classification method based on the Bayesian probability theorem, which assumes that the input variables are conditionally independent from each other, given the output variable. NBC performs well on small numbers of observations, automatically learns feature interactions and handles irrelevant features that are not required for prediction. Moreover, NBC is simple to implement, insensitive to noisy data and performs well in many domains (Stewart, 2002; Osmanbegovic & Suljic, 2012; Feng;Ding;Chen;& Lin, 2013; Soni & Vivek Kumar, 2018). A subsequent goal is to ascertain the viability of the predictive model(s) for use in early warning system in order to facilitate early identification of potential at-risk students, as well as the identification of trends and patterns to accommodate better interventions.

Accordingly, this study attempts to develop a model(s) with explanatory variables selected on the basis of our previous findings to predict student performance as well as to identify students who need support early in the semester.

1.2 Research goals and objectives

As stated previously, the objective of this research is to develop a validated machine learning based predictive models to predict student academic performance in programming to identify at-risk early in the course of study. The three objectives of this study were as follows:

i. Identify and select suitable data mining techniques to develop a mathematical model(s).
ii. Develop and validate the mathematical model(s) using the educational data collected from computer programming course(s) to
    a. Identify the factors that foster student learning performance in programming courses.
    b. Explore the course specific factors that influence academic performance.
    c. Predict or identify, at an early stage, the low performing students.
iii. Propose the developed model(s) as an early warning system to predict/identify at-risk students early in the semester and, in turn, potentially to inform both instructor and student.

1.3 Research questions (RQs)

Five research questions have been designed to address each research objective of this study, and the research questions are;

RQ1. Which feature selection techniques should be used to identify the influential factors that affect student learning and academic progress based on available academic data?

RQ2. How might a predictive model be developed and validated to predict performance in programming courses?

RQ3. What combination of predictors/independent variables yields the highest prediction accuracy to predict student academic performance?

RQ4. What percentage of academically at-risk students may be correctly identified by the models?

RQ5. How suitable are developed models for incorporation in early warning systems, for educators to identify students that need assistance in introductory programming courses?

In publications P1, P2 and P3 we examined the feature selection techniques for identifying the most relevant factors affecting student learning and academic performance, that contribute to research question RQ1. Specifically, these three publications focused mainly on identifying key factors that may serve as best predictors in predictive model construction. Moreover, in these publications, we employed various data mining techniques to select suitable features for further exploration in subsequent studies P4 and P5. Similarly, publications P4 and P5 present studies in relation to the research questions RQ2-RQ5. In addition, replication and extended studies have been conducted based on findings from P5, to confirm that models based on our prior studies may be deployed as early warning systems, in order to predict/identify students that requiring early assistance.

1.4 Scope of this thesis

Student academic performance can be affected by various factors. This thesis focused on developing a predictive model based on student academic data, collected via surveys, homework, demonstration, tutorials and mentoring session of a specific course to explore the unidentified patterns in order to identify the factors that influence student's learning and academic performance to predict their academic performance. In addition, student perceived prior programming knowledge and problem-solving skills were included in constructing predictive models based on prior studies P2 and P3. However, other psychological factors, such as self-esteem, self-regulated learning and emotional states were not included in constructing predictive models.

1.5 Significance of this thesis

This thesis is significant in further promoting technology-enhanced learning environment and enhancing personalised learning skills. The findings of this thesis will contribute towards learning and teaching of computer programming, which is vital in the context of computer science and IT curricula. The recommended approach derived from the results of this thesis may be applied at schools to improve student learning outcomes. Educators will be guided on what should be emphasised in the university curriculum to improve students' performance in computer programming courses.

The findings of this study will be helpful for educators, students and researchers in the following ways:

- Provide a predictive model with course specific factors that influence student's learning skills and academic performance will help educators to redefine their teaching methods and strategies in teaching programming courses.
- Provide a process to design and create a prediction model that predicts at-risk students who may face academic difficulty at early stage of the course will help educators to help them succeed.
- Provide suggestions to reallocate or tune the learning technologies that are in use to align with student's learning preferences based on identified influential factors from the defined model.
- Provide suggestions to foster student learning skills, self-efficacy and increase in academic achievement based on results of student's academic progress from the defined model.
- Assist the instructors to extract patterns of performance, areas of weakness or strength, and to identify students who need more attention than others.
- Deployment in other courses with similar goals.

1.6 Structure of this thesis

The rest of the thesis is organised as follows. Background of the study presents the theoretical foundations such as importance of learning analytics and educational data mining in predictive modelling, machine learning algorithms, and early warning systems relevant to this study (Chapter 2). The Related work section presents a literature survey of important previous work, conducted around prior knowledge, problem-solving skills, lecture attendance and formative assessment tasks, and their significance in relation to student final exam grades, predictive modelling and early warning systems (Chapter 3). Summary of publications section presents the summary of our published articles including results and contributions (Chapter 4). Research methodology section describes the methods used in the replication study conducted based on P5 for this thesis to find answers for our research question RQ3-RQ5 including the details about the courses and development of models (Chapter 5). Data analysis and results section presents the findings of the replication study conducted based on prior studies (Chapter 6), which I discuss in depth in discussion section including prior publications. Finally, conclusions, limitations and future work section presents our conclusions and limitations in terms of how well the foregoing research questions is answered, and we identify some related future work directions, to develop a more enhanced and innovative approach to teaching introductory programming (Chapter 7).

Chapter 2

Predictive Modelling: Learning Analytics, Educational Data Mining, Machine Learning, and Early Warning Systems

This chapter provides the background information related to the research publications by the author, and which have substantially contributed to chapters of thesis. The importance of learning analytics, educational data mining, machine learning and early warning systems, all related to predictive modelling, are presented in this chapter, as these topics form the basis of our prior studies as well as the replication study conducted for this thesis. The predictive model proposed in this thesis, to predict at-risk students early in the semester, was arrived at via analysis of introductory programming course data. The underlying concepts include Learning Analytics (LA), Educational Data Mining (EDM), Machine Learning (ML) and Early Warning Systems (EWSs). The chapter presents a background to these concepts to better situate the development of the proposed predictive model.

2.1 Learning analytics (LA) in predictive modelling

Learning Analytics (LA) is a composition of a set of techniques and algorithms that are used to measure, collect, analysis and extract results from data about learners and their contexts to directly support instructors and students (Pardo, 2014). In other words, LA is about learning, and is an emerging field that seeks to answer questions arising in contexts of teaching and learning, in order to enhance aspects of learning. The impact of educational technologies on student learning has offered new opportunities to gain useful insights into teaching and learning environments and demands the need of LA. For example, using student log and course performance data to predict student behaviors and subsequent learning outcomes is one of the most diverse areas within LA research. LA based predictive modelling with educational data mining techniques has become a core practice of educational researchers and largely with a focus on predicting student academic performance in education (West;Luzeckyj;Searle;Toohey;& Price, 2018). In this thesis, we are mainly concerned with student course performance and course entry survey data collected via ViLLE, a learning management system (LMS) to determine LA-based predictive modelling with data mining techniques.

In the wake of the Internet, student online learning activities and course performance are captured and stored as digital traces or log data to identify patterns of learning behaviors, via educational data mining techniques. However, simply identifying learning patterns of students does not guarantee success of an education practice. That is, “How do we positively use these identified learning patterns or information to impact instructors’ teaching practices and enrich students’ learning outcomes?” or “How might the captured data be utilized to derive models that are capable with predicting student learning outcomes that will occur in the future? And, “What kind of manual or automatic actions and solutions should be implemented in the learning setting from the source data collected as well as any identified patterns and predicted results?" It implies that, results of educational data mining need to be further analysed, in order to properly provide insights into teaching and learning. As such, our studies (P1-P5) have deployed different fields of LA by using educational data mining techniques, to identify answers for what happened? (Descriptive analytics: P1), why did it happen? (Diagnostic analytics: P2 and P3), what will happen? (Predictive analytics: P4 and P5) and how can we make it happen? (Prescriptive analytics: P5 and replication study results), in the field of CS education.

2.2 Educational data mining (EDM)

We now go through an introduction about EDM, which is applied in all our included publications and replication and extended study of this thesis. EDM is an important process to discover significant facts, unknown trends and patterns, and relationships in data that come from educational settings to understand student learning. The main goal of EDM community is to apply innovative data mining methods on educational data to discover hidden connections in order to achieve the goal of "enhancing educational practice". EDM is otherwise called as knowledge discovery in database (Mohamad & Tasir, 2013). Moreover, EDM is one of the prominent research fields of LA (Chatti;Dyckhoff;Schroeder;& Thus, 2012). EDM is an analytics process with advanced tools and technologies to develop methods to harness the educational data points and their intersections to identify patterns from that to reveal student behaviours, and subsequent learning outcomes for LA to create actionable intelligence in order to improve student learning. As such, EDM focuses on data analysis paradigms and LA focuses on human intervention. Notably, there were four major classes of EDM methods those frequently used by analytics in the field of education. They are prediction models, relationship mining, structure discovery, and discovery with models. In these, prediction models are very prominent in both EDM and LA communities (Baker & Inventado, 2014). For our study we used prediction models includes machine learning algorithm were explained in subsequent sections.

2.3 Predictive modelling

Predictive modeling is a process that uses statistics including machine learning algorithms with collected data and relevant predictor variables to predict future results. The process of developing predictive model is called predictive analytics. Predictive analytics in education uses statistical and machine learning algorithms to predict future events based on past educational data. The objective of predictive analytics in education is to predict the student performance, student retention, student enrollment, institutional progress and more based on the current and past student and institutional data in order to assist learners, instructors, course administrators and academic advisors to draw sense making decisions. For example, when the student learning outcomes is predicted at the initial stage (based on his/her past and current academic and or nonacademic data) then it would be easier for instructors to help students those predicted as low-motivated learners to alleviate their learning issues in order to reduce drop-out rates. Academic predictive models are developed by using the data, statistical and machine learning algorithms/techniques to provide answers for the questions that have been raised and unanswered in education. Predicting student performance in programming courses is a topic that has received much attention in computer science education for decades. Furthermore, collecting student learning process data via learning management systems

(LMS) such as Moodle, Blackboard, Canvas and ViLLE to understand the processes involved in student learning and the progress gained has also received much attention in the fields of EDM and LA.

Moreover, selecting the most efficient variables as predictors (called variable selection) in predictive models, determine the prediction accuracy and longevity of the model. Variable or feature selection is the process of selecting suitable subset of features that may serve as best predictors in a predictive model construction and to improve the results. This step is also important in machine learning as it helps in understanding data, reducing computation requirement, and better model interoperability (Chandrashekar & Sahin, 2014; Miao & Niu, 2016). Moreover, including unnecessary features in a model will influence the predictive performance of the model. Notably, 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 by using various variable selection techniques. That is, selecting a subset of relevant features is the most important process in predictive modelling. It also implies including unnecessary feature influence predictive performance of the model. The most common variable selection methods those widely used in research studies are; filter, wrapper and embedded methods. As such, our studies P1-P3 examined the factors that influence student performance in programming and studies P4 and P5 discussed the role of variable selection for predictive model development. Notably, P4 used filter method and P5 used wrapper method for variable selection.

2.4 Machine learning (ML) algorithm

ML is a branch of statistics or is a set of mathematical techniques that implemented on computer systems and provides the ability to those systems to learn from the given input (data) and experience to predict future outcomes (Morgan, 2018; Chio & Freeman, 2018). There are two types of machine learning algorithms used for development of predictive models. They are, supervised learning (regression or classification), and unsupervised learning (clustering) based algorithms. Generally predictive models fall into one of these three categories namely clustering or classification or regression depends on the nature of data and problem. There are many machine learning algorithms that widely used for predictive modelling depends on the nature of collected data and problem. Figure 2.1 shows how machine learning algorithms deployed on collected data for predictive analytics based on its nature.

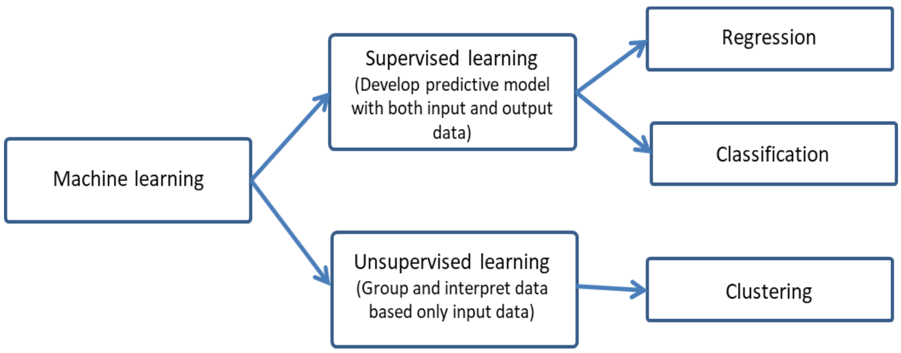

Figure 2.1: Machine learning algorithms on predictive model development.

Here, supervised learning approach deals with labeled data (data with meaningful label or classified with suitable tag) and unsupervised learning deals with unlabeled data (data with no labels or with many labels). For example, in unsupervised learning we use clustering technique to identify the patterns of the input data. Figure 2.2 shows the machine learning algorithm implemented computer works on unsupervised data to derive inferences from given input (K-means clustering algorithm).

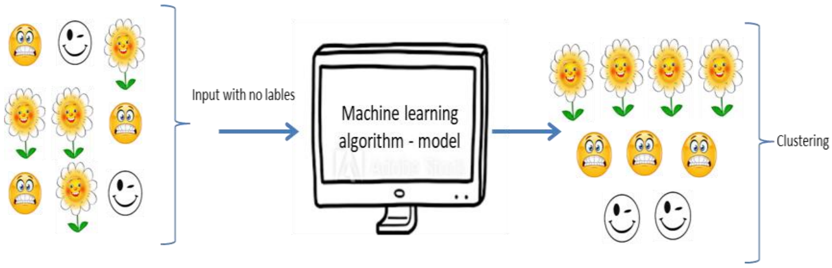

Figure 2.2: Unsupervised learning on data that have no labels for clustering.

In supervised learning we have machine learning algorithms for classification and regression. For example, Linear regression, Logistic regression, KNN, Naïve Bayes, Random forest are some common supervised learning algorithms widely used in predictive modelling with supervised learning (Liao;Zingaro;Laurenzano;Griswold;& Porter, 2016; Conijn;Snijders;& Kleingeld, 2017; Al-Shehri, et al., 2017; Francis & Babu, 2019). Table 2.1 shows extract of first year programming students' continuous assessment data and grade obtained in the year 2016 for supervised learning.

ID	PPK*	Homework	Demo	Final exam	Grade	Status
x1	2	100	40	98	5	QUALIFIED
x2	2	93.22	25.33	29	0	UNQUALIFIED
x3	1	99.83	97.33	67	2	UNQUALIFIED
x4	0	87.45	86	90	4	QUALIFIED
x5	2	100.00	92.66	100	5	QUALIFIED
..	..	..	..	..	..	..
* Prior programming knowledge (PPK)						

Table 2.1: data collected for supervised learning (classification or regression).

Regression models are used to predict continuous or ordered whole values (for example, student final exam scores). Classification models are used to predict discrete class labels (for example, student final course grades). As the collected data are tagged with unique labels (Table 2.2) the supervised learning based predictive model(s) can be developed.

Input/predictor variables	Output/predicted variable	Type of supervised learning problem	Machine learning algorithm (example)
Homework and demo	Final exam (continuous)	Regression	Linear regression / Random forest
Homework and demo	Status (qualified or unqualified)	Binary classification	Naïve Bayes / Logistic regression
Homework and demo scores	Grade (0 or 1 or 2 or 3 or 4 or 5)	Multiclass classification	Naïve Bayes / Neural networks / Random forest…

Table 2.2: Regression / classification based predictive model: Supervised learning.

This implies that selection of type of predictive model or implementation of learning algorithm is based on the nature of the dataset and output variable is in or set or the problem. However, it should be noted a classifier may predict a continuous value provided that a continuous value is in the form of a probability for a class label. Similarly, the regressor may predict discrete value provided the discrete value is in the form of an integer value. As noted, there are many machine learning algorithms that widely used for predictive modelling depends on the nature of collected data and problem. We deployed Support vector machine regression and Naïve Bayes algorithm classification based predictive models (supervised learning) in our prior studies P4, P5 and replication study of this thesis, respectively.

2.5 Early warning systems (EWS)

Academic early warning system (EWS) is a computerised system that designed to capture and analyse student data to identify student who need academic support, and to identify key factors that influence student retention and learning outcomes. The EWS acts as a student progress indicator, allowing educators use such information to support off-track students before they drop out or reach critical condition (P5). For example, Signals project from Purdue University, and Student Explorer from STEM academy are kinds of early warning systems designed with core of LA to identify students that need support and provides real-time feedback, interventions as early as possible (Pistilli;III;& Campbell, 2014; Krumm, Waddington, Teasley, & Lonn, 2014). These projects analyse data accumulated in LMS to identify student that need support and identify factors that impact academic advisor's decisions. As such, we introduced our validated models as early warning systems to predict at-risk students early in the semester and, in turn, potentially to inform both the instructor and student.

2.6 Summary

This chapter has highlighted the need for LA, EDM and feature selection to develop predictive models that typically include machine learning algorithm(s) and to heed the student engagement-based findings in order to improve student learning. We therefore deployed LA, EDM and machine learning algorithms in our publications P1-P5, which focused on development of statistical predictive models that uses data mining techniques and machine learning algorithms to predict student learning outcomes. Consequently, we presented research studies conducted around these topics in relation to introductory programming, the need for at-risk student identification, predictors used for model development, and validation and incorporation of our validated models as early warning systems in the next chapter.

Chapter 3

Computer programming courses, predictive factors and predictive models: Related work

This chapter provides further motivation for this thesis by presenting the important related work, within the context of key areas of focus, of relevance to our study. These include teaching and learning of introductory programming, problem-solving skills, prior programming knowledge, lecture attendance, formative and summative assessment tasks, predictive models and machine learning techniques, which are emphasised in the research questions.

3.1 Introductory programming

Computer programming is the process of writing set of commands that get executed by computers. Programming is a vital skill and a rewarding career for students who are interested in computer science and IT. It is claimed that learning to program improves student general problem-solving and thinking skills (Psycharis & Kallia, 2017; Yukselturk & Altiok, 2017). Therefore, introductory programming is emphasised as one of the recommended courses for non-computer science students at tertiary level. However, introductory programming is considered to be a major stumbling block for many students and many studies have reported the difficulties faced by novices when learning programming (Qian & Lehman, 2017; Luxton-Reilly, et al., 2018). The worldwide average, successful completion rate in introductory programming is 67.7%, with failure rates continuing to be the range of 28-32% ((Watson & Li, 2014; Bennedsen & Caspersen, 2019). Students enrolled in introductory programming courses often experience difficulties in grasping basic programming concepts and algorithms (Lister, et al., 2004). Novices struggle to understand programming concepts as they lack of clear mental models to relate to programming concepts (Moskal;Gasson;& Parsons, 2017) and novices often write code with misconceptions and syntax and logic errors (Ettles;Luxton-Reilly;& Denny, 2018; Zingaro, et al., 2018; Izu;Mirolo;& Weerasinghe, 2019). There are at least three reasons for this. First, computer programming courses require students to have a good understanding of programming concepts and meta-cognitive skills, such as problem solving and high-level thinking skills, in order to be proficient in programming (Uysal, 2014). Second, students must have the abstract thinking and logical principles in order to visualise and to solve real world problems in code form. Third, the programming proficiency of novice learners is dependent on the choice of the programming language that offered in introductory programming course. (Koulouri, Lauria, & Macredie, 2015). That is, the programming language that offered at introductory programming might impact the development of programming skills of novice learners. Hence, several studies have attempted to identify factors that contribute to ability in learning and success in programming, including but not limited to students' psychological and cognitive characteristics and study behaviour (Watson;Li;& Godwin, 2014; Lishinski, Yadav, Enbody, & Good, 2016; Lishinski;Yadav;& Enbody, 2017).

However, these studies need further verification due to inconsistencies in results obtained over a range of identified factors (Longi, 2016).

3.2 At-risk students and the need for predicting student academic performance

The phrase “at-risk students” is typically used in educational settings to refer a group of students who struggle with their studies or risk of failing academically or have higher probability of dropping out of school. They are usually low academic achievers, who need academic support from instructors and academic advisors. Increase in at-risk student numbers, course non-completion and student attrition rates, cause poor university outcomes, of concern to all stakeholders (students, instructors, course administrators, academic advisors and institutions) (Jia & Maloney, 2015). Programming is difficult for novices to learn and failure rates are high (Bennedsen & Caspersen, 2007; Silva, 2014). The need for early indicators of students becoming at risk has been explored, based on identified factors around student success/failure, so that early intervention strategies maybe deployed (Macfadyen & Dawson, 2010; Helal, et al., 2019; Liao;Zingaro;Alvarado;Griswold;& Porter, 2019).

Despite this concern, research studies emphasise the need for prediction of student academic performance for a number of reasons. First, predicting student academic performance is an important research endeavour at higher education level and highly valuable for instructors to execute timely interventions (Conijn;Snijders;& Kleingeld, 2017). Second, improving student learning and, increasing student success rates, are important and long term goals for educational institutions towards providing quality education (Asif, Merceron, & Pathan, 2015; Yassein;Helali;& Mohomad, 2017). Universities capture large volumes of digital educational data of their students to understand and address student success, retention and graduation rates to create actionable intelligence knowledge (Pistilli;III;& Campbell, 2014). However, transforming such large volumes of data into knowledge is challenging and, which requires enhanced predictive methods to transform those captured data into meaningful patterns to enrich student learning experiences (Asif, Merceron, & Pathan, 2015; Shahiri;Husain;& Rashid, 2015). Third, there are no clear metrics thus far to identify student retention (Pistilli;III;& Campbell, 2014). So, identifying key factors that influence student performance would help to predict at-risk students at an early stage, to minimise the drop-out rate and improve retention. Fourth, there is a substantial body of empirical literature on machine learning techniques-based predictive models (utilising data mining and learning analytics) for student performance and to identify students that need support (Ahadi, Lister, Haapala, & Vihavainen, 2015; Leppänen;Leinonen;Ihantola;& Hellas, 2017; Luxton-Reilly, et al., 2018). However, as student predictions are inconsistent in nature, robust models are needed, to accommodate learning data that changes over time and to deliver significant predictions.

3.3 Identifying predictors of student achievement

Several studies have been conducted to detect the factors that influence student learning outcomes, and which may be used to predict student academic performance (Astin, 1978; Longi, 2016; Luxton-Reilly, et al., 2018; Liao;Zingaro;Alvarado;Griswold;& Porter, 2019). Evans et al. listed 34 independent variables that that might be used to measure student understanding of programming concepts (Evans & Simkin, 1989). Also, studies have cited family causal factors, academic causal factors, and personal causal factors affect student academic performance (Aguiran;Lazo;& Salabat, 2014; Akar &

Altun, 2017). However, there is no concrete inventory that may be used as a possible predictor, as the results have often been inconsistent, and predictor variables used in these studies have varied from one context to another, with variations occurring in student cohort, cultural setting, class size and classroom and academic environments (Sharma & Shen, 2018). In addition, the data sources which are used in the aforementioned studies are often so complex that the predictor variables correlated in a complicated, non-linear way (Guo, Zhang, Xu, Shi, & Yang, 2015). Consequently, research seeks predictors that can produce consistent results on predicting student academic achievement, despite the contextual issues that impact student performance. So, for this thesis the predictor variables selected based on educational psychology and our prior studies to predict student performance in final programming exam despite contextual factors that affect student achievement for our model.

3.4 Prior programming knowledge (PPK) as a predictor of student performance

Prior knowledge is knowledge that can be defined as an individual's prior personal stock of information, skills, experiences, beliefs and memories. Prior knowledge is reported as an important variable in educational psychology research (Ausubel, Novak, & Hanesian, 1978) and has long been considered as one of the most important factors that influence student learning behaviours, experience and performance (Buskes & Belski, 2017; Adamopoulos, 2017; Tzu-ChiYang;Chen;& Y.Chen, 2018). Research studies related to student perceptions on prior knowledge in learning mathematics, programming and science courses reported that prior knowledge in topic is a factor of success (Hailikari;Nevgi;& Komulainen, 2007; Tafliovich, Campbell, & Petersen, 2013; Nivala;ParankoHans;Gruber;& Lehtinen, 2016). Students who have PPK perform better in programming than those who have no prior knowledge (Longi, 2016; Hsu & Plunkett, 2016; Kori;Pedaste;Leijen;& Tõnisson, 2016). However, few studies also claim that inaccurate prior programming knowledge may hinder new learning and raise misconceptions (Marling & Juedes, 2016). Furthermore, some students who had no PPK attained higher grades than students who had PPK in an introductory programming (Alexandron, Armoni, Gordon, & Harel, 2012). Despite these mixed results, PPK is often discussed and included in predictive models as an input variable (Longi, 2016; Grover, Pea, & Cooper, 2016). In addition, our prior study (Veerasamy, Daryl D'Souza, & Laakso, 2018) on the impact of PPK on lecture attendance and final programming exam confirmed that prior knowledge in programming influences student lecture attendance and final exam performance. Therefore, PPK was included in our study as one of the predictor variables of the model developed in this thesis.

3.5 Problem solving skills (PSS) as a predictor of student performance

Problem solving is a kind of effective thinking or a complex mental activity to find solutions for difficult or complex issues. Problem-solving skill (PSS) is a valuable skill, which needs to be acquired in learning and workplace to ensure success. Moreover, problem-solving skills are identified as one of the required "employability skills in the 21st century workplace", along with technical skills (Suarta;Suwintana;Sudhana;& Hariyanti, 2017). For example, to become a computer scientist it is necessary to have adequate knowledge in programming, practice in solving problems and designing systems (Kappelman;C.Jones;Johnson;R.Mclean;& Bonnme, 2016). As such, problem-solving is a basic required skill for students. Several studies refer to PSS as a cognitive and prerequisite factor for student achievement in many courses (Behjoo, 2013; Bester, 2014). PSS and self-efficacy are related and therefore PSS might influence student's academic self-efficacy in learning programming (Erözkan, 2014). However, Lishinski et al. reported that student problem-solving ability did not correlate significantly with student performance in multiple-choice exams (Lishinski, Yadav, Enbody, & Good, 2016). Despite these mixed results, in higher education a significant effort is directed towards the development of metacognitive and PSS in order to improve students' thinking and problem-solving, to ensure success in learning and in the workplace. For example, pedagogical approaches such as collaborative learning, and problem-based learning were implemented to enhance student programming PSS in novice programming learning (Uysal, 2014; Jackson, Lawson, Diack, Khosravi, & Vincent-Finley, 2016; Bawamohiddin & Razali, 2017) suggesting that PSS are essential for learning and has a connection with student learning abilities. In addition, our study (P3) on relationship between PSS and student performance in introductory programming courses revealed that students with PSS achieved better score in final programming exam than students with no PSS (Veerasamy;D'Souza;Lindén;& Laakso, 2018). This implies that PSS and learning programming are interrelated and student PSS can be used to determine student learning and performance in programming courses.

3.6 Lecture attendance (LEA) as a predictor of student performance

Lecture is a traditional and continuous to be a one of the effective teaching methods in most universities at present. Students who attend lectures regularly are likely to succeed in academics (Jover & Ramírez, 2018). The relationship between student lecture attendance (LEA) and academic performance is widely researched (Narula & Nagar, 2013; Lukkarinen;koivukangas;& Seppälä, 2016; Kassarnig, et al., 2018). Regular attendance in lecture got a positive impact on student learning despite the availability of online resources (Alexander & Hicks, 2016). LEA and student academic performance are positively correlated in introductory programming courses (Bai;Ole;& Akkaladevi, 2018). However, Chapin reported that low or high attendance in lecture did not impact student final grades of first year and second year university psychology students (Chapin, 2018). On the other hand, Kassaring et al. measured the LEA of 100 university technical students and concluded that early and consistent LEA strongly correlates with students' academic performance (Kassarnig;Bjerre-Nielsen;Mones;Lehmann;& Lassen, 2017). Despite these mixed results, LEA is used as one of the predictors in machine learning based models for predictive analytics to predict student academic performance in various courses (Mueen;Zafar;& Manzoor, 2016; Rix;Dewhurst;Cooke;& Newell, 2018; Gatsheni & Katambwa, 2018). However, our study (Veerasamy, et al., 2016) revealed that formal LEA and novice student's final programming exam performance was negatively correlated. As such, this thesis did not use student LEA as one of the predictive variables.

3.7 Formative assessment tasks (FA) as a predictor of student performance

Assessment tasks represent a wide range of activities including homework, essays, group work assignments, oral presentations, case studies, online quizzes and tests and written examinations. Assessment plays an important role in student learning and influences student achievement (Gaal & Ridder, 2013). The purpose of assessment is to measure whether a student has achieved intended learning outcomes for a study module (Gibbs, 2010). For example, formative assessment tasks (FA) conducted by academics during a course, typically aligned with the course syllabus requirements and which reflects the desired student learning, in terms of their incremental progress in learning. It partially determines students' final performances. It is claimed that the practice of FA is rooted in Bloom's concept of "mastery learning", an instructional strategy and educational philosophy that adopts the use of assessments to measure student's learning outcomes (S. & Hastings, 1971). FA is aimed at stimulating and directing student learning (Timmers;AmberWalraven;& P.Veldkamp, 2015) and plays a significant role in the student learning process (Gibbs, 2010). Assessing students with frequent assessments increases study motivation reduces procrastination and enhances academic performance (Gibbs, 2010; Gaal & Ridder, 2013). Students aware that completing FA (for example homework) may lead to improved final grades (VanDeGrift, 2015). Furthermore, educators use FA such as homework as predictors to identify where students are struggling in order to assist them and to address their problems (Gibbs, 2010).

Homework (HE) is formative assessment that is given to students to complete at home or outside class times to test their comprehension of the subject (Rajoo & Veloo, 2015). There are three types of HE: practice homework (study for tests, essays), preparation homework (demo exercises, group work), and extension homework (project work, case studies) prepared and delivered to promote student learning. HE impacts student performance and is important for student achievement (Rajoo & Veloo, 2015; Planchard;Daniel;Maroo;Mishra;& McLean, 2015). Moreover, additional HE has a significant impact on student achievement in exams (Eren & Henderson, 2008). However, formative assessments have no significant impact on final exam scores and failure rates, although it improves overall performance in lab work (Gratchev & Balasubramaniam, 2012; Gaal & Ridder, 2013). A meta-analysis by Fan et al. on HE and student achievement in mathematics and science revealed that HE has insignificant positive relationship with academic achievement (Fan;Xu;Cai;He;& Fan, 2017). On the other hand, our prior study (P1) on impact of continuous summative assessments on student achievement in programming courses concluded that HE and demo exercises have a positive significant correlation with student achievement in final programming exams. However, the correlation coefficient value varied year-to-year though the relationship between the selected assessment tasks and student final programming exam performance was significantly positive (Veerasamy, et al., 2016). The aforementioned studies revealed that FA plays a vital role in student learning and achievement. Moreover, the early weeks of formative assessment results provide good opportunities to partially assess student learning outcomes and to identify at-risk students. As such, in our studies including P4, P5 and the replication study, we included performance in ongoing assessment tasks as predictor variables, based on our prior study, P1, for model development and to identify at-risk students in programming.

3.8 Predictive modelling for student academic performance

Predictive modelling comes under the category of predictive analytics. It is a kind of mathematical model which may employ classifiers or regressors to formulate a statistical model. In education predictive modelling is generally used in predicting student performance in a course and to identify students at risk of course failure. There have been several studies conducted to develop predictive models employing various data mining algorithms for predicting student performance in computing education (Bergin, Mooney, Ghent, & Quille, 2015; Devasia, P, & Hegde, 2016). Furthermore, predictive models may be used in an early warning system to identify students who need support by facilitating the use of a variety of strategies to communicate with selected at-risk students and provide them pathways for improving their performances (Krumm, Waddington, Teasley, & Lonn, 2014).

Several studies examined the effectiveness of different machine learning algorithms to select the suitable classification machine learning algorithms for predictive models (Dekker;Pechenizkiy;& Vleeshouwers, 2009; Perez;Castellanos;& Correal, 2018; Hussain;Zhu;Zhang;Abidi;& Ali, 2018). However, it is not clear, yet which machine learning algorithm is preferable in this context. For example, Devasia et al. employed Naïve Bayesian's classification to predict final grades of computer science students and found that it was more accurate when compared with other data mining methods, including linear regression, decision tree, and neural networks (Devasia, P, & Hegde, 2016). However, Bergin et al. found that there were no significant statistical differences between the prediction accuracy of Naïve Bayes and Logistic regression, Support vector machine, Artificial neural network and Decision trees data mining techniques, in predicting introductory programming student performance, even though Naïve Bayes was found to have the highest prediction accuracy (Bergin, Mooney, Ghent, & Quille, 2015). Other studies reported that Support vector machine, when used for model generation and validation, achieved the best performance in predicting success over other classification and regression-based algorithms (Bydžovská, 2016; Liao, et al., 2019). Liao et al. deployed Logistic regression model to perform binary classification for predicting student performance in multiple CS courses. They stated that Logistic regression was selected for model development due to its simplicity and ability to work well with a small number of input features (Liao;Zingaro;Alvarado;Griswold;& Porter, 2019).

In addition, these aforementioned studies used various model evaluation techniques to validate model performance, in order to determine how well these models would perform on unknown data. For example, Borra et al. measured the prediction error of the model by employing estimators such as Leave-one-out, parametric and non-parametric Bootstrap, as well as cross-validation methods, and reported that the repeated 10-fold cross-validation estimator and the parametric bootstrap estimators performed better on estimating the prediction error of the model, than leave-one-out and hold out estimators (Borra & Ciaccio, 2010). Many studies deployed confusion matrix (CF) for measuring the prediction accuracy of classification algorithm-based models (Mueen;Zafar;& Manzoor, 2016; Liao, et al., 2019). Notably, area under the curve is a probability curve (AUC) measure, used in several studies to determine how well the model predicts the classes best (Thai-Nghe;Busche;& Schmidt-Thieme, 2009; Yukselturk;Ozekes;& Türel, 2014; Anderson;Boodhwani;& Baker, 2019). For example, Liao et al. analysed the value of different data sources for predicting student performance in CS courses and determined most valuable data sources based on AUC results (in compliance with AUC scores) of each data source used as predictors (Liao;Zingaro;Alvarado;Griswold;& Porter, 2019).

There have been studies explored the factors that influence the predictive accuracy of the model (Austin & Tu, 2004; Kattan, 2011; Austin & Steyerberg, 2015). The accuracy of prediction models might vary from dataset to dataset on the type of classification. For example, the dataset which contains small portion of students fail or dropout and the vast majority pass is called imbalanced dataset. The model developed with imbalanced data may produce overoptimistic results (Novianti;Jong;Roes;& Eijkemans, 2015). 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 (Derksen & Keselman, 1992). Our study (Veerasamy;D'Souza;Lindén;& Laakso, 2019) found that although the overall success of the model is significant, model overfitting and, lack of predictors might affect predictive accuracy of the model.

From these studies the following points emerged. First, selection of type of machine learning algorithm(s) is based on the nature of the data and target variables (whose values are to be modelled and predicted by other variables) is in or set or the problem. Second, adding more predictor variables does not necessarily help improve prediction accuracy of the model. Moreover, inclusion of highly correlated predictor variables in a model might cause the "multicollinearity" or variance inflation factor (Huang & Fang, 2013). Third, it is important to know how well the model(s) will perform for the future or unknown data. Fourth, the performance of the predictive model depends on the sample size.

3.9 Predictive models as academic early warning systems (EWS)

EWS is an alert tool and designed to support both instructors and students. It facilitates the instructors to monitor student attendance, engagement, and course assessment performance at certain intervals in visual form to explore new patterns for decision making. These early alert systems have been used quite extensively in many educational intuitions to identify at-risk student, provide support and improve retention and graduation rates (Baepler & Murdoch, 2010; Jokhan;Sharma;& Singh, 2018). Notably, mining LMS (Blackboard, Moodle) data to develop an early warning system for course administrators, instructors and students is a significant active field of learning analytics research since last decade (Macfadyen & Dawson, 2010). Arnold et al. developed course signals, a student success system that analyse data collected by instructional tools and LMSs such as Blackboard Vista to produce course early warning signs and provides intervention to learners who may not be performing to the best of their abilities before they reach a critical point (Arnold & Pistilli, 2012). Similarly, Krumm et al. designed "Student Explorer" - EWS with a core of learning analytics to support STEM (Science, Technology, Engineering and Mathematics) students in a university. This EWS is designed to analyse the accumulated LMS data to identify students that need academic support and to identify factors that influence academic advisor's decisions (Krumm, Waddington, Teasley, & Lonn, 2014). Notably, EWS selectively used for freshmen courses targeting specific student populations such as first-year students rather than for all students (Simons, 2011). Some other studies investigated student attitudes towards these EWS and how they prefer to receive these early warning tools results in the course of their studies to improve their academic performance (Atif;Richards;& Bilgin, 2015; Roberts;Howell;Seaman;& Gibson, 2016). However, most EWSs designed heavily rely on student demographic and or LMS access data but not on performance data (Kuzilek;Hlosta;Herrmannova;& Zdrahal, 2015; Marbouti, Diefes-Dux, & Madhavan, 2016). In addition, Most of the EWSs developed based on continuous-summative data but not including cognitive and psychological factors. As such, this thesis developed a predictive model as EWS with variables that include student performance data and cognitive factors such as prior knowledge and problem-solving skills.

3.10 Summary

The need for early indicators to identify students at-risk is important to establish and facilitate timely interventions. Several studies have attempted to identify such early indicators for identifying students in need of support in programming. However, there is no concrete inventory that may be used as a possible predictor. In addition, student predictions are inconsistent, and require models that are able to accommodate learning data changes over time to produce consistent results on predicting student performance. Hence, we used predictor variables that accounted for variations in academic setting to predict student performance in the final programming exam despite contextual factors that affect student achievement for our model. As such, this study included non-collinear predictor variables that may have better explanatory predictive power, in order to build a possible balanced model (denoted parsimonious models), in turn, to attain feasible prediction accuracy in predicting student final exam grades in introductory programming. Second, this study used the classification-based algorithm, Naïve Bayes, to build models with predictors, selected on the basis of our previous findings (P1-P5) as well as the contributions to research questions, presented in the next chapter. Third, we deployed a K-fold cross-validation technique to evaluate the predictive performance of models for validation and testing. In addition, a confusion matrix was used to measure the prediction sensitivity, specificity, balanced accuracy, and AUC values, to compare the predictive models developed for this thesis, such comparison, allowed for predictive quality to be determined of models and to determine how well they would perform on unknown data, and to then propose an appropriate early warning system.

Chapter 4

Summary of publications

This chapter presents a summary of publications which have contributed to this thesis. They present studies involving a range of factors that ostensibly influence student performance, such as lecture attendance, homework, and prior programming experience, for example. The studies form a strategic and cohesive pursuit of factors to include in parsimonious predictive models, to better predict students at risk of failing the final exam.

The first three research articles (P1, P2, and P3) present a list of data mining techniques those were used in order to identify the influential factors that affect student learning and academic performance in programming courses. The next two articles (P4 and P5) present the development and validation of mathematical models using the selected features based on prior studies to predict low performance students at early stage of the course and propose one of those developed model(s) as an early warning system.

4.1 P1: The Impact of Lecture Attendance on Exams for Novice Programming Students

Summary: This paper examines the influence of lecture attendance and continuous assessment tasks on student performance in the final examination. Lecture attendance is widely considered as one of the key determinants of student learning and academic performance in many courses. Similarly, several studies alluded to formative assessment tasks as one of the important factors that influence student achievement in exams. However, this assumption needs to be tested due to the radical impact of educational technologies on student learning and performance. Moreover, there are contexts in which students mostly work from a distance and rarely attend classes at institutions. It is therefore essential to measure the impact of lecture attendance, continuous summative assessment tasks on final exam performance. In this study, correlation coefficient and multiple regression analysis (Mann, 2009) were implemented to assess the influence of lecture attendance on novice student learning and performance in programming courses.

Results and contribution to research questions: The correlation results for lecture attendance on formative and summative assessment tasks revealed that lecture attendance and assessment outcomes are weakly correlated. However, the correlation and multiple regression results for formative assessment tasks on final exam performance suggested that formative assessment tasks might be considered as predictor variables to identify student achievement in final exam. The data and results of this study might be used for further research to identify the learning preferences of novice programming students in order to enhance learner-centered classrooms. This publication contributes to research question (RQ1) by giving quantitative results on identifying the factors that foster's student learning performance in computer programming courses.

4.2 P2: The impact of prior programming knowledge on lecture attendance and final exam

Summary: This publication examines the similar problem as P1 but with cognitive factor dataset using various statistical methods. This publication reports the results of the impact of student prior knowledge in programming on lecture attendance and on subsequent final programming performance in a university level programming course. This analysis attempted to answer the research question “Why do some students skip lecture sessions yet, do well in the final exam?” This question was raised based on P1 results and it is identified that students entering our first-year programming course with varied programming knowledge and experience which could have influenced their lecture attendance and academic performance. Therefore, this study analysed the impact of prior programming knowledge on lecture attendance and final programming exam by using statistical and visualisation techniques. The Shapiro-Wilk test, Spearman’s rank correlation coefficient, multiple regression, Kruskal-Wallis, and Bonferroni correction tests were used to examine the student data (Ghasemi & Zahediasl, 2012; Mann, Nonparametric Methods, 2009).

Results and contribution to research questions: The study delivered mixed results. The Kruskal-Wallis and Bonferroni correction test (multiple comparison tests) results suggest that students who have prior programming knowledge will also have poor lecture attendance. Similarly, the multiple comparison test results revealed that students with high prior programming knowledge achieved higher scores in the final programming exam than student with no prior programming knowledge. In addition, the multiple regression results suggest that, student prior programming knowledge affect student lecture attendance and final exam performance. However, lecture attendance did not have any significant impact on student final exam performance. As such, this publication concludes that class attendance may not be considered as one of the factors that influence student performance. However, prior programming knowledge is significantly a better predictor to use to predict final exam scores in programming courses. This publication contributes to research questions RQ1, and RQ2 partially in order to use student prior knowledge as one of the predictor variables for model development and validation.

4.3 P3: Relationship between perceived problem-solving skills and academic performance of novice learners in introductory programming courses

Summary: This publication explored the influence of cognitive factor that foster’s student learning performance in programming courses in order to use it as input variable for predictive modelling. This study focused to answer the research question “Why is learning to program easier for some than the others?” The research reported here aimed to determine whether student perceived problem-solving skills is relevant to student performance in learning programming. This is because research in computer science education highlighted that problem solving is a valuable and desirable skills for students. Many novice students lack problem solving skills and have difficulties in utilising key programming concepts to express in their code. As such, this study explored the influence of student perceived problem-solving skills on formative and summative assessment tasks performance by using quantitative analysis.

Results and contribution to research questions: The Spearman’s rank correlation coefficient results revealed that students who have poor problem-solving skills might perform poorly in formative and summative assessment tasks. In addition, the multiple comparison test results revealed that effective problem solvers might perform better in the final exam than poor problem solvers. Furthermore, from these study results the following points emerged. First, it is possible to categorise students based on problem-solving skills, to explore student constructivists learning improvements. Second, although poor problem solvers performed similarly to moderate and effective problem solvers in formative assessment tasks, they failed to achieve high scores in the final exam due to lack of problem-solving transferability skills. Therefore, attention should be paid to align the formative and summative assessments in order to improve transferability skills. This publication contributes to research question RQ1 by giving quantitative results on identifying the course specific factors that foster student learning performance in computer programming courses for predictive modelling.

4.4 P4: Prediction of Student Final Exam Performance in an Introductory Programming Course: Development and Validation of the Use of a Support Vector Machine-Regression Model

Summary: In P4, the challenge in establishing valid predictive models was studied. This publication presents the support vector machine regression model to determine if prior programming knowledge and completion of selected continuous summative assessment tasks might be suitable predictors of examination performance. The features for predictive modelling were selected based on past research studies (P1 and P2), learning theories, and filter methods such as multiple regression. The developed predictive model was validated by using K-fold cross-validation technique.

Results and contribution to research questions: The results revealed that overall prediction accuracy of the model is moderate. However, predictions on identifying at-risk students are neither high nor low and that raised the following questions (i) What factors might have impacted the prediction accuracy of the model developed? and (ii) How to improve the prediction accuracy of the model in future? The possible answers for these questions were discussed in the publication in order to get more optimal tuning parameters to improve the model performance. This publication contributes to research questions RQ2 and RQ4 on developing and validating a predictive model for prediction of student performance and identification of student that need support.

4.5 P5: Predictive Models as Early Warning Systems: A Bayesian classification model to identify at-risk students of programming

Summary: In P5, the development and validation of parsimonious predictive models was studied. This publication presents the Naïve Bayes multiclass classification models to determine if student perceived problem-solving skills, prior knowledge in programming and completion of selected continuous summative assessment tasks might be suitable predictors of final exam grades. The features for predictive modelling were selected based on our prior studies (P1-P4). In addition, wrapper method was deployed to evaluate and select the combinations of features yields the highest prediction accuracy to predict student academic performance. Fifteen models with various combinations of selected features were developed and tested in P5. The objective of P5 was to answer the research questions RQ3, RQ4, and RQ5.

Results and contribution to research questions: The K-fold cross-validation results of P5 revealed that the overall prediction accuracy on identifying student final exam grades and identifying at-risk students were moderate and good. The results of P5 persuaded us to propose a generic model that can be deployed for other programming and non-programming courses, if the goal of the instructor is to predict student performance early in the semester.

4.6 Replication study results

Summary: This study was conducted to systematically analyse and verify our previous studies using data collected in the years 2016-2018 from two different introductory programming courses. This study is a replication and extension of our prior study P5. As such, similar research methodology (explained in section 5) was applied to answer our research questions RQ3-RQ5.

Results and contributions to research questions: The unknown data test results of this study shown that; it is possible to predict student that need support in the early weeks of the semester and re-answered our research questions RQ3, RQ4 and RQ5. The results of the models might be used as early warning signs and incorporated as early warning systems for instructors via ViLLE in visual form to provide intervention to learners before they reach critical point.

4.7 Contributions of the author

This thesis has sourced its content from the afore-mentioned five manuscripts (four of which have been published). These publications, submitted by the author, have independently addressed a range of factors that affect student final exam performance, using data mining techniques. The outcomes of the individual studies subsequently led to the development and validation of mathematical models using the outcomes from these prior studies to predict, at early stages during the course, low performing students with the overall aim of proposing an appropriate model(s) for incorporation in early warning systems. The student data used in all these studies (P1-P5 and replication study) was collected via ViLLE and with the help of ViLLE research team members (Peter Larsson, Erno Lokkila, Erkki Kaila, Teemu Rajala, and Einary Kurvinen). Details of contributions associated with each manuscript appear below.

The P1 is the first manuscript of the study, which explored the impact of LEA on novice student performance on in programming exams. I was the main author of this article; statistical analysis was done with the help of Mr. Rolf Lindén, and Erkki Kaila, and writing was done with the help of other authors Daryl D'Souza, Mikko-Jussi Laakso and Tapio Salakoski.

The P2 article is an extension of our prior article P1, which reports the results of impact of PPK on LEA and on subsequent final programming exam performance in a university level introductory programming course. It was a quantitative study and with the help of VilLE research team I was able to conduct ViLLE based entry survey to collect and analyse student data in the academic years 2012-2014 for this manuscript preparation. Research methodology was defined with the help of Rolf Lindén and the reporting was done with the help of Daryl D'Souza and Mikko-Jussi Laakso who are co-authors of this paper.

The P3 article is a joint effort by me, Daryl D'Souza, Rolf Lindén and Mikko-Jussi Laakso. The data for this article was collected via ViLLE. Mr. Erno-Lokkila, instructor for *Algorithms and Programming* helped me to conduct PSI survey online for introductory programming courses. This article presents the relationship between PSI and academic performance of novice programming students. Rolf Lindén and I analysed the data using EDM while, written content was contributed to by Daryl D'Souza and Mikko-Jussi Laakso.

The P4 article was written by me and it was a preliminary exploratory study to understand how to develop predictive models for programming courses. The model development and selection of features was done by me, R coding and selection of machine learning algorithm was determined by Rolf Lindén and complete reporting was done with the help of co-authors Daryl D'Souza and Mikko-Jussi Laakso.

The P5 study was an extension of our prior study P4 and was written to answer our research questions RQ3, RQ4, and RQ5 of this thesis. This article was written with the help of ViLLE research team (helped to collect data for the study), and co-authors of this article.

The replication-extension study explained in this thesis (Sections 5 to 7) was written by me with the support of my supervisors Mikko-Jussi Laakso, Daryl D'Souza and Tapio Salakoski.

Chapter 5

Developing and validating predictive models: Research methodology

Our study was set up as a replication and extended study to verify our previous study P5 using larger dataset with different structure in order to know how well the models we developed and validated in P5 will perform on future or on unknown data. As such, this chapter presents our research methodology, including the research design, data collection, variables used as predictors, data pre-processing, and predictive model development procedures. It should be noted the instruments and features described in this replication study were taken from our prior studies (Veerasamy, et al., 2016; Veerasamy, Daryl D'Souza, & Laakso, 2018; Veerasamy;D'Souza;Lindén;& Laakso, 2018; Veerasamy;D'Souza;Lindén;& Laakso, 2019) and (P5). For example, PSI and PPK survey questionnaire and details defined in P1-P3, confusion matrix and followed by measures such as sensitivity and specificity used in this study were already defined in P5.

5.1 Research methodology

The overall goal of this research was to develop a model with reliable predictors for incorporation in academic early warning systems. This chapter describes how the predictor variables were identified, and the predictive models developed, using 15 x 2 courses combination of predictors. Three semesters (2016, 2017, and 2018) of student academic data for the courses *Introduction to Programming* and *Algorithms and Programming* were used for this study. Data was collected via ViLLE and, SPSS (IBM, 2013) and R (Team, 2013) software were used for statistical analysis. Table 5.1 presents the dataset collected initially for the replication-extension study.

*Dataset /course name [2016 + 2017 + 2018]	Introduction to Programming	Algorithms and Programming
Total number of students enrolled for the course	93+94+102=289	248+258+311=817
Total number of students completed PSI survey	65+68+66=199	230+222+266=718
Total number of students completed course entry-PPK survey	80+81+92=253	213+239+287=739
Total number of students attended final exam (FE)	66+68+70=204	174+175+224=573
*The data for the different course deliveries was not combined but used separately for different phases of predictive model development, validation and final testing.		

Table 5.1: Initial data collected for the study (2016, 2017, and 2018).

In total, over the three years, there were 289 students enrolled in the *Introduction to Programming* (Table 5.1). The initial data collected for model development in the year

2016 was 93. Of these only 54 students participated course entry surveys (PSI and PPK), and completed HE and DE exercises and FE, used to develop a model with K-fold cross-validation. The initial data collected for model validation in the year 2017 was 94. Of these 68 students completed the PSI and PPK surveys, HE and DE exercises and FE, used as sample data to verify the model's performance in line with our study objectives. Similarly, the initial dataset collected for model testing (unknown data) in the year 2018 was 102. Of these 20 students secured <=25% in selected FA in the first two weeks are identified as at-risk students for visualisation. Of remaining data of 2018, 63 students' data that completed the PSI and PPK surveys, HE and DE and FE, used as unknown data to test the final model fit for generalisation. Table 5.2 presents the dataset used for the development, validation and testing of predictive models for the course *Introduction to Programming*. The breakdown of the participating 185 was 54 in the year 2016, 68 in the year 2017 and 63 in the year 2018 for *Introduction to Programming*.

Introduction to Programming	**Actual**	**No. of students attended PSI, PPK survey, completed assessment tasks and attended FE**	**Dataset for training, validation and testing**
2016	93	54	54 (Training dataset)
2017	94	68	68 (Validation dataset)
2018	102	63	63* (Unknown dataset)
* Students that secured <= 25% in the first two weeks in the year 2018 were visualised as at-risk students and excluded from unknown dataset.			

Table 5.2: Dataset used for prediction models: *Introduction to Programming*.

Similarly, in total, over the three years, there were 817 students enrolled in the *Algorithms and Programming* (Table 5.1). The initial data collected for model development in the year 2016 was numbered 248. Of these only 170 students participated in the course entry surveys (PSI and PPK), and completed HE and TT exercises, and FE, used to develop a model with K-fold cross-validation. The initial data collected for model validation in the year 2017 was numbered 258. Of these 145 students completed the PSI and PPK surveys, HE and DE exercises and FE, used as sample data to verify the model's performance. Similarly, the initial dataset collected for model testing (unknown data) was numbered 311 in the year 2018. Of these 32 students secured <=25% in selected FA in the first two weeks are identified as at-risk students for visualisation. However, note that FE is not compulsory in *Algorithms and Programming* and registration to attend FE is allowed until the last lecture week of the course. Hence, the number of students appearing for FE in the year 2018 is unknown, which persuaded us not to use the 2018 data (students secured >=25% in selected FA) for testing, as our developed model may not fit with the course FE conducting polices. As such, 2016 data was used for model development (K-fold cross-validation) and 2017 data used for model testing (unknown data) in *Algorithms and Programming*. Table 5.3 presents the dataset used for the development and validation of predictive models for the course *Algorithms and Programming*. The breakdown of the participating 315 was 170 in the year 2016 and 145 in the year 2017 for *Algorithms and Programming*.

Algorithms and Programming	Actual	No. of students attended PSI, PPK survey, completed assessment tasks and attended FE	Dataset for training, validation and testing
2016	248	170	170 (Training dataset)
2017	258	145	145 (Validation/ test dataset)
2018	322	* Students that secured <= 25% in the first two weeks of the year 2018 were visualised as at-risk students.	

Table 5.3: Dataset used for prediction models: *Algorithms and Programming*.

5.2 Overview of the course

5.2.1 *Introduction to Programming* (INT)
INT course is taught in Java programming language. It is offered once a year to students from different disciplines. This course is offered in English and the duration of the course is 12 weeks. The course comprises of 24-26 hours of lectures, 20 hours of demonstration sessions and 10 hours for practice exam and discussion of project or assignment work, over an 11-12-week semester (Veerasamy, et al., 2016). The FE is mandatory, and students must secure at least 50% to pass the course. However, to be eligible to sit for the FE students must previously have secured at least 50% in homework, 40% in demo exercises and expected to submit the project work before FE. The final course grade is calculated based on scores secured in the FE as well as bonus points obtained via selected formative assessment tasks and lecture attendance.

5.2.2 *Algorithms and Programming* (ALG)
ALG course presents introductory programming using the Python programming language as a teaching vehicle. This course is offered in Finnish and the duration of the course is 8 weeks. The course comprises of 28 hours of lectures, 14 hours of tutorial and 8 hours of demonstration sessions, over an 8-week semester. The final grade for this course is calculated based on scores received in selected formative assessment tasks and or FE (Veerasamy, Daryl D'Souza, & Laakso, 2018). Student may get 1-2 course grade points at the maximum based on his/her performance in selected assessment tasks. To obtain course grade 3-5 student must attend FE and the final grades calculated based on scores received in FE including bonus points obtained from lecture attendance, and selected assessment tasks scores. However, student must have secured at least 50% in selected formative assessment tasks in order to sit for FE. The final course grade is calculated based on the scores received in the FE as well as bonus points calculated from lecture attendance and selected formative assessment tasks.

Both courses are designed for novice programming students and use ViLLE as the LMS/e-learning tool to support technology enhanced classes. There was no significant variation among student demographics, course periods, assignments, exams, and instructor in both courses.

5.2.3 ViLLE
ViLLE is mainly used for programming students, to deliver and manage course content, such as lecture notes, formative and summative assessment tasks for programming

students. It manages manually graded assignments and automated tasks, such as lecture attendance, demonstrations, file submission, study journals and course assignments (Veerasamy, Daryl D'Souza, & Laakso, 2018).

5.3 Description of predictor variables

For this study two surveys were conducted at the beginning of the semester for self-assessment of problem-solving skills and prior programming knowledge denoted as PSI and PPK respectively.

5.3.1 Problem-solving skills (PSI)

For this study the questionnaire developed by (Heppner, 1982) was used to collect student perceived PSS based on our prior study results (Veerasamy;D'Souza;Lindén;& Laakso, 2018). This PSI was used in various longitudinal studies to measure student general PSS in programming courses to identify differences between gender and their general PSI, improve programming skills and to enhance learners' PSS (Yurdugül & Aşkar, 2013; Uysal, 2014; Özen, 2016). Moreover, this measure can be applied to teenagers and adults. The questionnaire contains 32 closed Likert format questions with a 6-point Likert scale. In addition, we ran the Cronbach's Alpha test to measure the PSS reliability, which yielded 0.835, indicating a high level of internal consistency with the data collected, for our scale. Henceforth for brevity we drop the abbreviation PSS and use PSI instead for "Student perceived problem-solving skills".

5.3.2 Prior programming knowledge (PPK)

To collect PPK, a course entry survey was conducted at the early stage of course session. ViLLE was used to create and collect student PPK. The survey for PPK contained 3-point survey questions to ensure that each question had an optimum number of response categories and a number beyond which there was no further improvement in terms of the distinction between the rated items and those used in our prior study (Veerasamy, Daryl D'Souza, & Laakso, 2018). Both PSI & PPK survey questionnaires provided in the Appendix (Appendices 9.1-9.3).

5.3.3 Homework exercises (HE) assessment

HE is set as weekly formative assessments for ALG and INT provided for a total of 8 weeks and over 10 weeks, respectively. These exercises are offered to students via ViLLE and allow them to electronically submit their answers. Submitted answers for HE is automatically graded via ViLLE. The possible total raw score for HE for INT and ALG was 890 and 317, respectively.

5.3.4 Demo exercises (DE) assessment

DE for INT was provided to students weekly, for 10 weeks, and bi-weekly (from the 4th week onwards) for ALG, via ViLLE throughout the semester. Students are expected to prepare solutions for DE exercises at home and present their solutions in designated DE sessions. In a DE session, all student solutions are discussed, and a few students are selected randomly via ViLLE, to demonstrate their answers to the entire class. No marks are awarded for class demonstrations. However, students who complete the DE are instructed to enter their responses in the lecturer's computer to record the number of DEs completed by them. The marks for DE were calculated by ViLLE based on their registered responses in the lecturer's computer (Veerasamy, Daryl D'Souza, & Laakso, 2018). The possible total DE for INT and ALG was 750 and 300 respectively. However, DE for ALG was delivered to students via ViLLE after three weeks. Hence, this study did not include DE to predict student FE grades for ALG.

5.3.5 Tutorial exercises (TT) assessment

Each set of TTs for ALG was provided to students weekly for a total of 8 weeks. In a tutorial session students are given coding exercises via ViLLE to work online in the classroom. Students are allowed to submit their answers online on their own or in a group, and submitted exercises are automatically graded by ViLLE. However, a few coding exercises are manually graded by lecturer, with scores entered into ViLLE. The possible total TT for ALG is 650. INT course does not offer tutorials for students.

Both HE and DE are hurdles for INT with students having to attain at least 50% or over HE and 40% or over DE in order to pass these components and the course. Similarly, all HE, TT, and DE are hurdles for ALG and students must secure at least 50% in each component and ALG course students must complete the end semester online-assignment in ViLLE, to be eligible to sit for the FE. Both TT and DE sessions are conducted in the classroom and partially supervised and assisted by the lecturer.

5.3.6 Final exam (FE)

The FE is a summative assessment task conducted at the end of each course, electronically submitted via ViLLE. The FE is a hurdle for INT and student must secure at least 50% to pass the hurdle and to be eligible for a course grade. However, FE is not compulsory for ALG to pass the course, provided students attain at least 80% over all the selected assessment components to receive the maximum of two credit points and course grade 2. To obtain grades from 3 to 5 students must secure at least 50% in the assessment components and should get at least 62% in the FE (Table 5.4). The possible total FE score for INT and ALG is 100 and 90, respectively.

5.3.7 Final exam grade (FEG)

The FEG for the course is calculated based on FE scores. Table 5.4 shows the grade calculation in detail that used for this study to predict FEG for both courses.

INT		**ALG**	
FE marks	**Grade***	**FE marks**	**Grade***
0 to 49	0 (FAIL)	0 to 44	0 (FAIL)
50 to 59	1	45 to 55	1
60 to 69	2	56 to 66	2
70 to 79	3	67 to 77	3
80 to 92	4	78 to 88	4
93 +	5	89 +	5
* The actual grades 0 and 1 are considered as "at-risk" and denoted as ZERO; Grades 2 and 3 as "good" and denoted as ONE and grades 4 and 5 as "very good" and denoted as TWO for this study.			

Table 5.4: Grading criterion table-INT and ALG.

5.4 Data collection and pre-processing

The main objective of this study is to identify students who needed support in the early weeks of the semester, for the instructor to intervene, in order to improve student learning. As such, data for the cognitive variables: PSI and PPK were collected in the first week of the semester for both courses INT and ALG, for the years 2016, 2017 and 2018. As noted earlier, the course duration for INT and ALG were 12 and 8 weeks respectively. So, the formative assessment task (HE and DE) data for INT was collected after two weeks (Week 2 data), after four weeks (Week 4 data), and after six weeks (Week 6) of the semesters (2016, 2017 and 2018), for model development, validation, and testing. Similarly, formative assessments (HE and TT) data for ALG was collected after two weeks (Week 2 data), after three weeks (Week 3 data), and after four weeks (Week 4 data) of the semesters (2016 and 2017) for model development and validation/testing.

Data pre-processing is an important step in predictive model development, as incomplete, noisy, discrepancies or inconsistent data potentially affects predictive model performance. As such, the data collected via ViLLE was pre-processed. This study used SPSS and R software to pre-process the data in order to transform the raw data into a more understandable format (IBM, 2013; Team, 2013). First, the actual HE and DE/TT scores (for the first six weeks of the term, for all years) were transformed into percentages. The scaled dataset was stored as .xlsx/csv files to implement the developed predictive model, based on these pre-processed datasets. Table 5.5 shows the variables with the description and values stored as dataset for predictive analytics (extracted from P5).

Data pre-processing for predictive modelling			
Variable	**Description**	**Type**	**Values**
HE	Homework	Continuous	The actual HE, DE/TT secured converted into percentage
DE	Demo exercise	Continuous	
TT	Tutorial exercise	Continuous	
PSI	Problem-solving skills	Discrete	Integer values in between 32 and 192
PPK	Prior programming knowledge	Categorical	0→ No knowledge; 1→ Basic knowledge 2→ Good knowledge
FE	Final exam	Discrete	Integer values in between 0 and 100 (INT) / 0 and 90 (ALG)
FEG	Final exam grade	Categorical	Calculated from FE scores (Table 5.4)

Table 5.5: Variables with the description and values collected and stored as dataset for predictive modelling (P5).

The FEG for the courses was calculated from FE scores (Table 5.4) in order to maintain consistency between selected predictor variables and the output variable. The pre-processed datasets collected in the year 2016 were used to develop a set of machine learning algorithm based predictive models. The datasets collected in the years 2017 and 2018 were then employed to validate and test these developed predictive models. It should be noted that, data imputation was not used as imputing missing data can lead to biased feature estimates. Table 5.6 shows the calculated grade wise distribution data of INT for the years 2016-2018 and ALG for the years 2016-2017 for training (10-fold cross-validation), validation and unknown data testing for generalisation.

In this replication-extension study, we defined students that secured grades 0 (<50%: INT, <45%: ALG) or 1 (<60%: INT, <56%: ALG) in FE as at-risk. This is because; students that secure a passing grade may likely not to succeed in subsequent courses. As such, the actual grades 0 and 1 are considered as at-risk for this study and defined as grade “ZERO” (Table 5.6).

Final exam grade (FEG) *At-risk	**INT (Number of students)**			**ALG (Number of students)**	
	2016	2017	2018	2016	2017
*ZERO = Zero + One	21	16	29	44	28
ONE = Two + Three	9	21	12	54	44
TWO = Four + Five	24	31	22	72	73

Table 5.6: Grade wise distribution calculated from FE scores for INT and ALG.

5.5 Predictive model development, validation, and testing

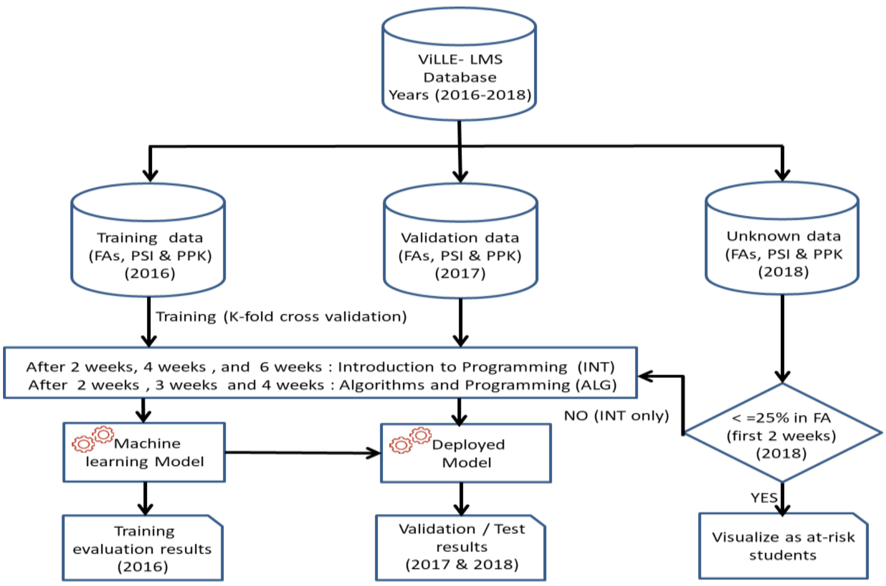

Figure 5.1: Modelling framework of replication study.

5.5.1 Criteria used for measuring prediction accuracy of models

The modelling framework used for this study is based on prior (Veerasamy;D'Souza;Lindén;& Laakso, 2019) and (P5). Student performance in FA (HE and DE/TT), PPK and PSI might act as early warning indicators for identifying students at-risk of course failure. In addition, the predictive model was developed with the supervised learning approach to excavate patterns of performance from assessment and other data, areas of weakness or strength, and to predict grades or learning outcomes. The intent behind this model was to identify students who needed attention and to refer them to relevant support activities before they reached critical points. This study deployed the Naïve Bayes classification algorithm with K-fold cross-validation to predict students' final exam grades. Figure 5.1 shows the modelling framework of this replication study derived from P5.

The classification accuracies of the developed models were evaluated based on a confusion matrix (CF) computed via R coding. CF is a table that presents a summary of prediction results for binary and multi-class classification-based models (Fawcett, 2006). The table is prepared with four different combinations of measures for predicted and actual values. CF is mainly used to compute predictive model prediction sensitivity, specificity, positive and negative predicted values and balanced accuracies, in order to weigh and compare the prediction accuracy of the developed models. Table 5.7 shows the skeleton of multiclass classification problem-based confusion matrix table used for this study.

<table>
<tr><td></td><td colspan="4">Predicted</td></tr>
<tr><td rowspan="4">Actual</td><td></td><td>ZERO
(At-risk)</td><td>ONE
(Good)</td><td>TWO
(Very good)</td></tr>
<tr><td>ZERO</td><td>TP</td><td colspan="2">FN</td></tr>
<tr><td>ONE</td><td rowspan="2">FP</td><td colspan="2" rowspan="2">TN</td></tr>
<tr><td>TWO</td></tr>
</table>

Table 5.7: Confusion matrix table for performance measurement of models.

True Positive (TP): This means that the predicted positive class and the actual positive class are the same. In this study, the TP value represents the number of at-risk students (grade 0) who are correctly identified by the model.

False Positive (FP): This means that the predicted positive class and the actual positive class are not the same. In this study, the FP value represents the number of not-at-risk students (grades 1 and 2) who are incorrectly identified as at-risk students (grade 0) by the model.

True Negative (TN): This means that the predicted negative class and the actual negative class are the same. In this study, the TN value represents the number of not-at-risk students (grades 1 and 2) who are correctly identified by the model.

False Negative (FN): This means that the predicted negative class and the actual negative class are not the same. In this study, FN represents the number of at-risk students (grade 0) who are incorrectly identified as not-at-risk students (grades 1 and 2) by the model.

Model's at-risk prediction accuracy sensitivity (ATSE): This denotes the proportion of actual positive classes that got predicted as positive by the model. In this study, the ATSE value represents the percentage of at-risk students who are correctly identified by the model. The model's ATSE is calculated as;

$$ATSE = \frac{TP}{TP + FN} \times 100$$

Model's at-risk prediction accuracy specificity (ATSP): This denotes the proportion of actual negative classes that got predicted as negative. In this study, the ATSP value represents the percentage of not-at-risk students who are correctly identified by the model. The model's ATSP is calculated as;

$$ATSP = \frac{TN}{TN + FP} \times 100$$

Positive predictive value (PPV): The PPV measures the proportion of actual positives that are correctly identified. In this study, the PPV value represents the probability of actual at-risk students who would be correctly identified by the model. The PPV is measured by calculating number of actual at-risk students who were correctly identified as grade "0" (TP) by dividing the total number students predicted as at-risk (TP + FP) by the model. Then, the result is multiplied by 100 to get the PPV for the model.

Negative predictive value (NPV): The NPV measures the proportion of actual negatives that are correctly identified. In this study, the NPV value represents the probability of actual not-at-risk students that would be correctly identified by the model. The NPV is measured by calculating the number of actual not-at-risk students correctly identified as not attaining grade "0" (TN) by dividing the total number students predicted as not-at-risk (FN + TN) by the model.

Balanced accuracy (BAC): This measure the average accuracy obtained from each class in the model. In this study, BAC represents the overall probability that a student will be correctly classified by the model. It is calculated as,

$$BAC = \frac{(TP + TN)}{(TP + FN + FP + TN} \times 100$$

Model's overall classification accuracy (MAC): It denotes the overall model classification accuracy. Here, TPs represents the total number of both at-risk and not-at-risk students correctly identified by the model. In this study, MAC represents the model prediction accuracy in percentage and is calculated as,

$$MAC = \frac{Total\ number\ of\ TPs}{Total\ number\ of\ TPs + Total\ number\ of\ FPs} \times 100$$

Area under the curve (AUC): The AUC is a performance measurement for binary or multiclass classifiers. The AUC value lies between 0.5 to 1 where 0.5 denotes a bad classifier and 1 denotes an excellent classifier. In this study, we determined above 0.5 for AUC as a good model classifier (Fawcett, 2006). Furthermore, we determine the model's classification performance based on the model's MAC, ATSE, and ATSP and, in compliance with high AUC scores (closer to 1.0)

Ensemble method of classification: This is a method of combining the decisions/predictions from multiple models of same machine learning or different machine learning algorithms of same model to improve the overall prediction performance. In this study, we combined the at-risk predictions of models in the years 2017 and 2018 based on training and validation results for ALG and INT, respectively. Majority voting technique was applied to obtain the final output at-risk prediction to compute the at-risk prediction accuracy of the models tested for this replication-extension study.

5.5.2 Feature selection for model development

Model#	Feature with model equation	Type	Course
#1	PSI → FEG	Cognitive variables	INT
#2	PPK → FEG		
#3	PSI, PPK → FEG		
#4	HE→ FEG	Formative assessment tasks	
#5	DE → FEG		
#6	HE, DE → FEG		
#7	PSI, HE → FEG	Cognitive variables and formative assessment tasks	
#8	PSI, DE → FEG		
#9	PSI, HE, DE → FEG		
#10	PSI, PPK, HE → FEG		
#11	PSI, PPK, DE → FEG		
#12	PPK, HE → FEG		
#13	PPK, DE → FEG		
#14	PPK, HE, DE → FEG		
#15	PSI, PPK, HE, DE → FEG		
#16	PSI → FEG	Cognitive variables	ALG
#17	PPK → FEG		
#18	PSI, PPK → FEG		
#19	HE→ FEG	Formative assessment tasks	
#20	TT → FEG		
#21	HE, TT → FEG		
#22	PSI, HE → FEG	Cognitive variables and formative assessment tasks	
#23	PSI, TT → FEG		
#24	PSI, HE, TT → FEG		
#25	PSI, PPK, HE → FEG		
#26	PSI, PPK, TT → FEG		
#27	PPK, HE → FEG		
#28	PPK, TT → FEG		
#29	PPK, HE, TT → FEG		
#30	PSI, PPK, HE, TT → FEG		

Table 5.8: The models developed for feature selection: Naive Bayes classification (P5).

In order to measure how accurately the selected variables were able to predict student FEGs, and to identify students that needed support, 2 courses x 15 predictive models were developed, with the following combinations of predictor variables to measure the differences between predictive capabilities of these models. Table 5.8 shows the models developed for feature selection.

In addition, one of the objectives of this study was selecting a model(s) with a suitable subset of features yielding higher prediction accuracies, to use in early warning systems. In order to evaluate the prediction accuracy of the models, we used 10-fold cross-validation to ensure that the training and testing sets (year 2016) and validation sets (year 2017) contain sufficient variation to arrive at unbiased results. In turn, this would avoid overfitting and to establish how well the model generalizes to unknown data (year 2018). We used the wrapper method (forward selection) to determine whether adding a specific feature would statistically improve the predictive performance of the model (Li, et al., 2017). In addition, the process was continued until all available variables were successively added to a model, to identify the best set of variables for model development. The prediction accuracy of each of the 30 predictive models was examined by calculating the overall model prediction accuracy, the at-risk student prediction accuracy sensitivity and specificity, and area under the curve score (ROC curve), for each model. The following prediction accuracy measures were applied via R coding, to evaluate the performance of all models (in training, validation and testing) to answer our research questions.

Models #1-#3 and #16-#18 were developed using cognitive features as input variables to predict FEG for both courses. Models #4-#6 and #19-#21 were developed using formative assessment tasks as input variables to predict FEG for both courses. Models #7-#15 and #22-#30 were developed using both formative assessment tasks and cognitive factors as predictor variables to predict student FEG for both courses. The models (#1-#2, and #4-#5) and (#16 -#17, and #19-#20) were developed with single feature for INT and ALG to examine the MAC, ATSE, ATSP, BAC, and AUC (for multiclass) results of those models in order to identify the most valuable predictors for model development respectively. In addition, AUC for all classes and at-risk class versus all other classes measured to determine which of the models developed predicts the at-risk classes best.

For this study, the prediction accuracy on identifying at-risk and not-at-risk values (MAC, ATSE, ATSP, PPV, NPV, BAC in compliance with AUC scores >0.5) below 50% is considered as poor; 50% - 69% as moderate; and 70% and above as good.

5.6 Summary

This chapter presented the research methodology used to collect data and conduct a replication study to address the thesis research questions developed. The dataset collected in the years 2016, deployed for development and the datasets collected in the years 2017 and 2018, were deployed as validation and unknown data, respectively. Our study included two surveys, for self-assessment of PSI and PPK, and these surveys were conducted via ViLLE at the start of the semester. Two courses x 15 predictive models were developed with combinations of FAs (HE and DE) and cognitive variables (PSI and PPK) as predictors for feature selection. CF was mainly used to evaluate classification accuracies of the developed models. In next chapter we present the results of models developed, validated and tested including the influences of predictors that may serve as best predictors.

Chapter 6

Performance of predictive models: data analysis and results

This chapter presents the results of our replication-extension study, which was based on the research methodology presented in the previous chapter. The results presented include the effects of relevant parameters of the predictive models, and validation and testing of developed models**.**

6.1 Feature selection results

15 models x 3 terms for INT and 15 models x 2 terms for ALG were developed to determine the importance of predictors, to potentially serve as best predictors in a predictive model construction in programming. Prediction accuracy results of the models were tabulated and provided in the Appendices 9.4-9.5. Models with higher prediction accuracies in compliance with AUC scores were selected for further analysis.

6.1.1 Models with a single feature as predictor (training, validation, and testing) for INT (After Week 2 / Week 4 / Week 6): Models #1-#2 and #4-#5.

As noted, use of unnecessary features in a model will influence the predictive performance of the model. As such, the models #1-#2 and #4-#5 were developed, validated, and tested with a single feature for INT, to identify the single feature that most influences the model performance. The mean prediction accuracies (MAC, ATSE, and BAC) of DE, and HE computed over Week 2, Week 4, and Week 6 for the years 2016, 2017, and 2018 to determine the single feature that most influences the model accuracy. Figures 6.1, 6.2, and 6.3 present the average prediction accuracies of HE, DE and, PSI and PPK on predicting FEG in INT, using these variables in turn as single feature model predictors.

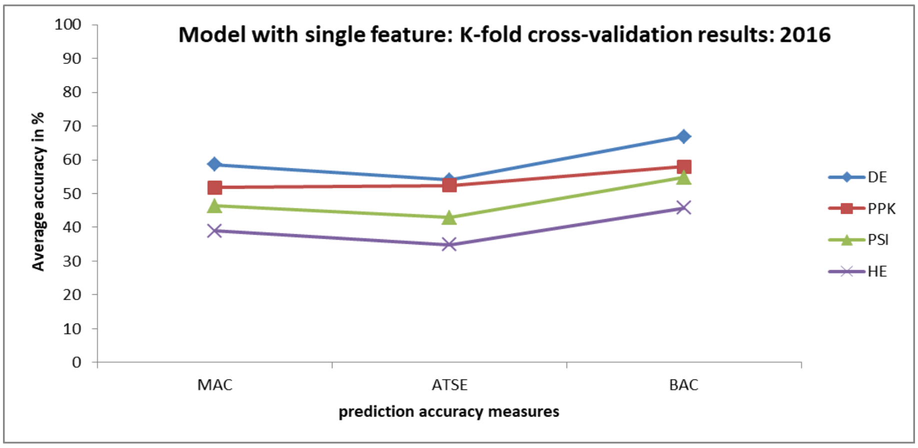

Figure 6.1: K-fold cross-validation results (2016): HE, DE, PSI, and PPK on FEG.

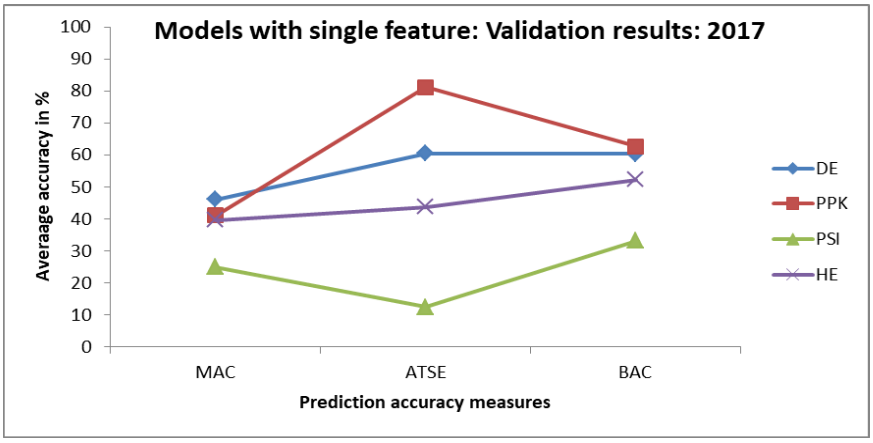

Figure 6.2: Validation results (2017): HE, DE, PSI, and PPK on FEG.

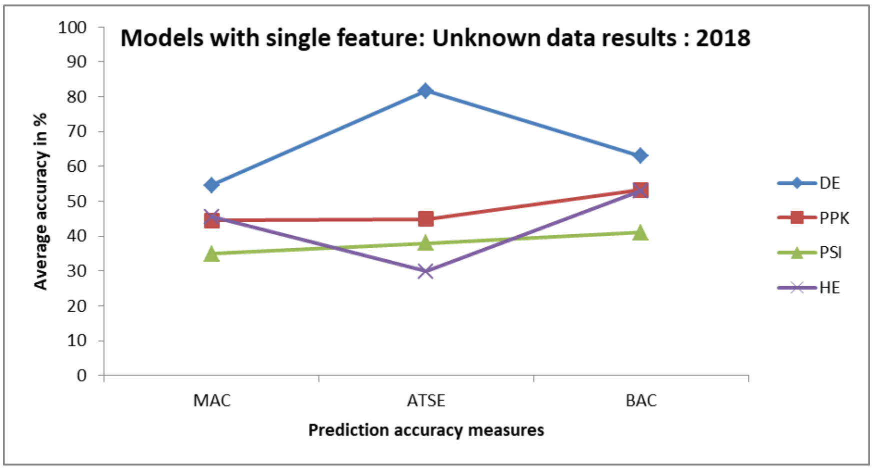

Figure 6.3: Unknown data results (2018): HE, DE, PSI, and PPK on FEG.

The models developed, validated, and tested with DE only as predictor (#5) at different early week study periods for INT had higher prediction accuracies in predicting FEG in compliance with AUC scores in between >0.50 and <0.70 in compare to models those developed with other features as single predictor. The models developed and tested with HE as single predictor (#4) had lowest prediction accuracies with insignificant AUC scores (between 0.46 and 0.55) (Figures 6.1-6.3). On the other hand, models developed with PSI or PPK only as predictor return mixed results. Models with PPK as predictor (#2) had nearly moderate prediction accuracies (training, validation, and testing) in compliance with AUC scores (between 0.56 and 0.58). However, models with other cognitive variable PSI only as predictor had moderate prediction accuracies (BAC) in training (Figures 6.1 and 6.3) but returned poor MAC, ATSE and BAC on validation and testing although AUC scores were moderate (in between 0.55 and 0.60) (Figure 6.2).

This implies that models developed with the combination of DE, PSI and PPK may yield higher prediction accuracies compare to models developed with other combination of features.

6.1.2 Models with a single feature as predictor (training and testing) for ALG (After Week 2 / Week 3 / Week 4): Models #16-#17 and #19-#20.

Similarly, models #16-17 and #19-#20 developed, tested with single predictor for ALG revealed that models developed and tested with PSI or TT only as predictor (#16 or #20) had higher prediction accuracies (in compliance with at-risk AUC scores in between 0.51 and 0.66) on identifying student FEG (with low false positives) in compare to models #17 and #19 developed with other features PPK and HE, respectively. Figures 6.4 and 6.5 present the models' average prediction accuracies (Week 2, Week 3, and Week 4) of selected formative assessments, and cognitive variables for the year 2016 and 2017 (training and testing).

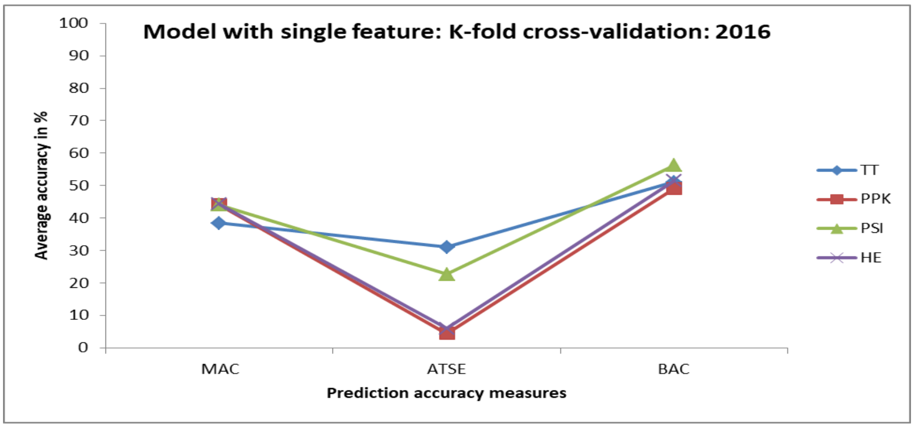

Figure 6.4: K-fold cross-validation (2016): HE, TT, PSI, and PPK on FEG

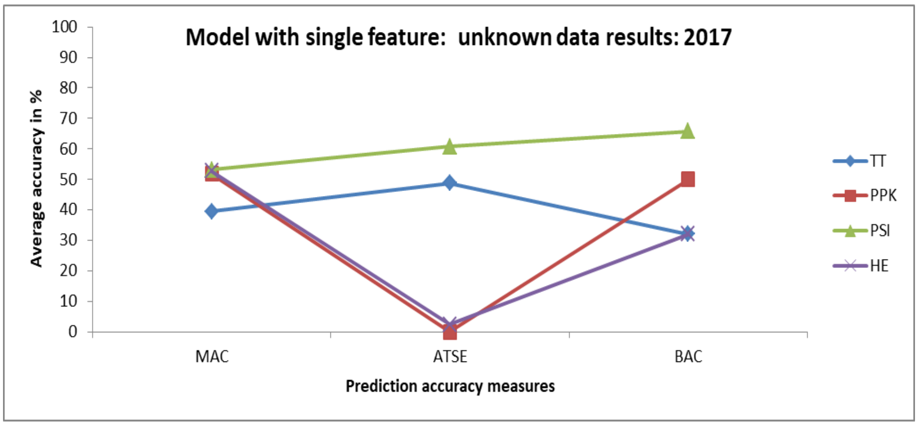

Figure 6.5: Unknown data results (2017): HE, TT, PSI, and PPK on FEG

In addition, the model developed with PPK or HE as predictor returned poor prediction accuracies (ATSE) (0% and 4%) on identifying at-risk students in compare to models developed with PSI (61%) or TT (49%) only as predictors.

6.1.3 Models with cognitive features (K-fold, validation, and testing) only as predictors for INT and ALG: Models #3 and #18.

The predictive models developed with cognitive variables only (PPK, PSI) as predictors were employed in the beginning of the course period (first week) to identify students in need of support, before second week of the semester. Figures 6.6 and 6.7 present the average prediction accuracies of the models with PSI and PPK only as predictors for INT and ALG respectively.

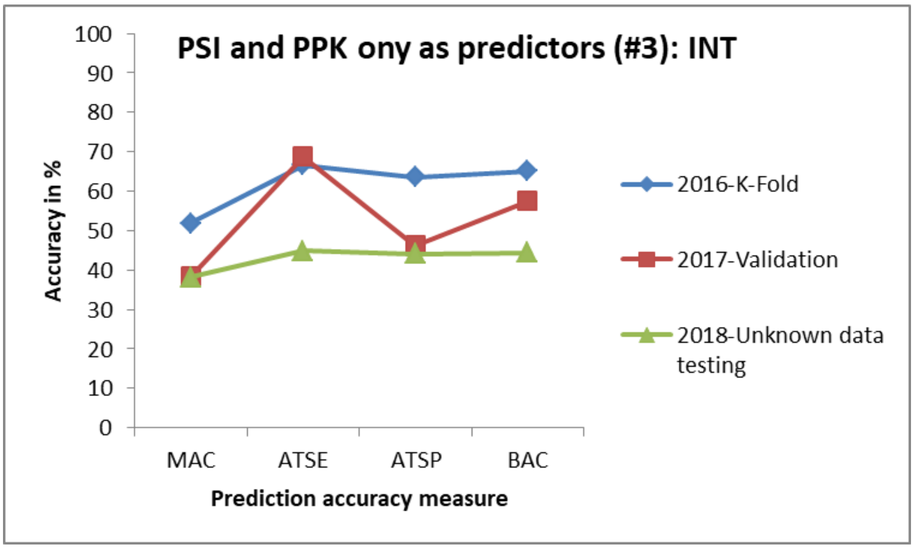

Figure 6.6: Cognitive features only as predictors (INT)

The predictive model (#3) developed and validated with cognitive features for INT had good significant prediction accuracies on identifying at-risk students (ATSE) in compliance with AUC scores (between 0.56 and 0.62) early in the course period. However, unknown data test results on model #3 had poor prediction accuracies on identifying at-risk (ATSE: 45%) and not-at-risk students (ATSP: 44%) in compliance with AUC score 0.58 for INT. In addition, the MAC of validation (38%) and unknown data (38%) were poor, although K-fold on test set yielded moderate MAC (52%) on identifying student FEG (Figure 6.6).

Similarly, the predictive model #18 with PSI and PPK as predictors developed and tested for ALG had poor MAC (45% and 46%). On average, the BAC of K-fold cross-validation results on identifying both at-risk and not-at-risk students was moderate (51%) but with insignificant AUC scores 0.47. On the other hand, the BAC on unknown data testing was good in compliance with moderate AUC score (0.52) (Figure 6.7).

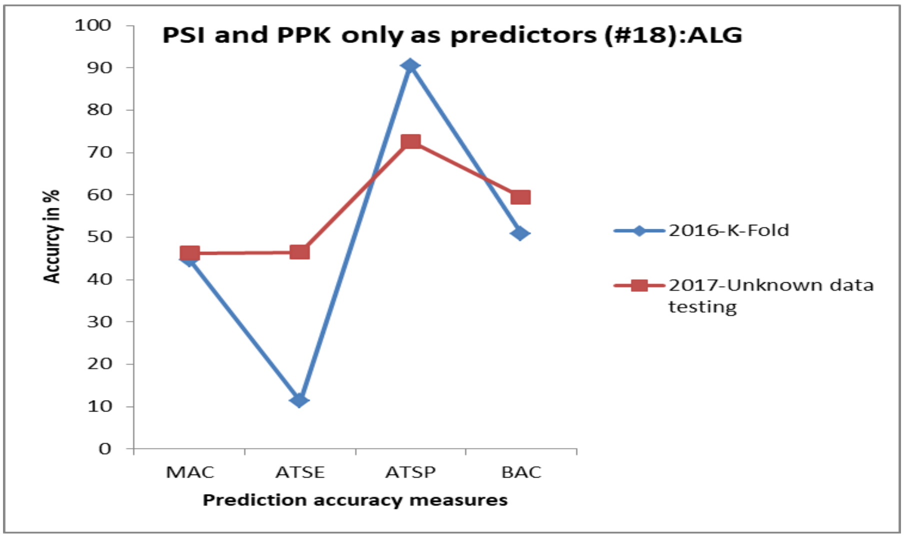

Figure 6.7: Cognitive features only as predictors (ALG).

6.1.4 Models with formative assessments only as predictors (K-fold/validation and testing) for INT and ALG: Models #6 and #2.

The predictive models developed with formative assessments only as predictors for INT were employed after the 2nd week, 4th week and 6th week of the course because the models required student HE and DE scores as inputs (#6). Similarly, the model developed with formative assessments HE and TT only as predictors for ALG (#21) were employed after 2nd week, 3rd Week, and 4th week, respectively. Figure 6.8 presents the average prediction accuracies (average of Week 2, 4, and 6 prediction accuracies) of model (#6) developed (2016), validated (2017), and tested (2018) for INT.

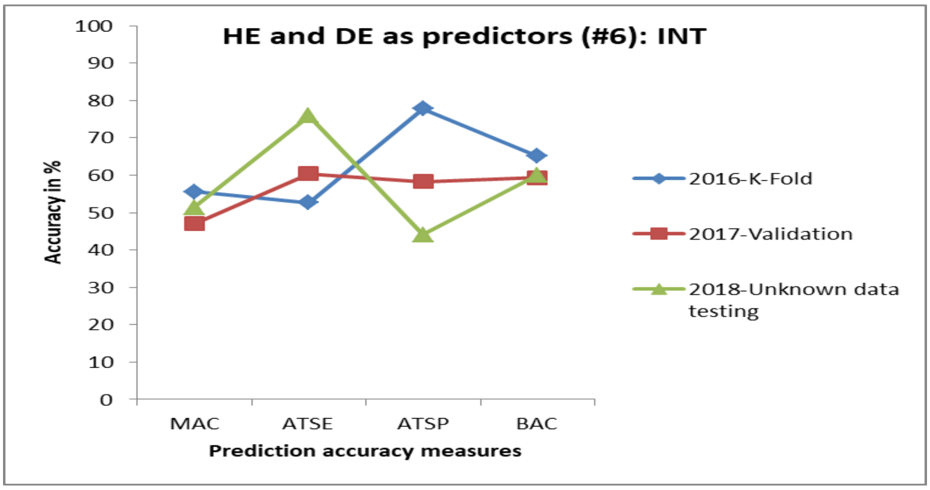

Figure 6.8: Formative assessments only as predictors (INT).

On average, the model with formative assessments only as predictors had moderate and good BAC (2016: 66%, 2017: 59%, and 2018: 60%) on identifying at-risk and not at-risk students in compliance with AUC scores (between 0.53 and 0.60) for INT. The overall MAC on K-fold cross-validation (56%) and unknown data (51%) for INT was moderate, although MAC on validation was poor (47%).

Similarly, Figure 6.9 presents the average prediction accuracies (average of Week 2, 3, and 4 prediction accuracies) of model (#21) developed (K-fold) and tested for ALG. The model with formative assessments only as predictors had poor prediction accuracies in identifying at-risk students in compliance with insignificant AUC scores (between 0.46 and 0.51: bad classifier) for ALG.

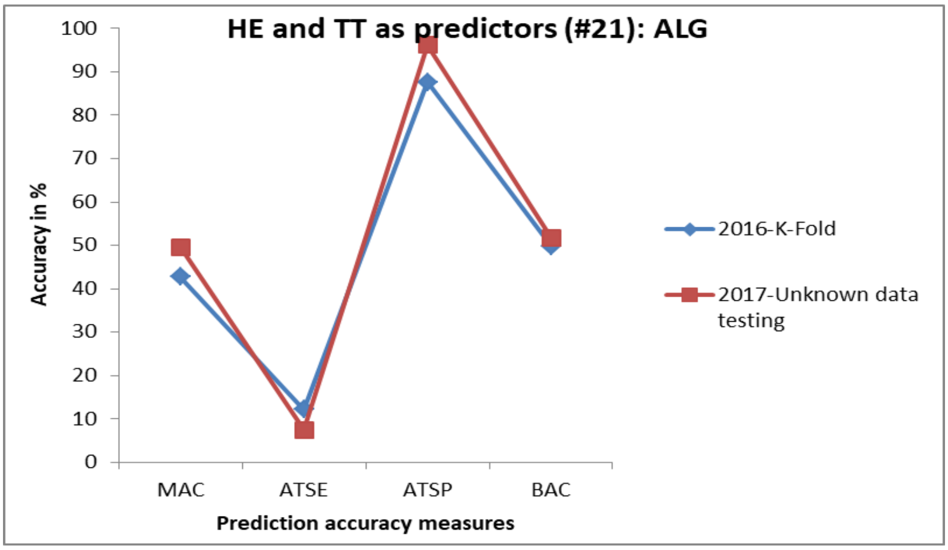

Figure 6.9: Formative assessments only as predictors (ALG).

6.2 Predicting student final programming performance

One of the objectives of this study was to identify the combination of predictor/independent variables that yields the highest prediction accuracy to predict student performance (RQ3). As such, models #6-#15 and #21-#30 were developed with various combinations of selected features for INT and ALG respectively. Of these, models that had higher prediction accuracies in compliance with AUC scores after Week 2, Week 4, and Week 6 were selected for further analysis.

Figure 6.10 shows the models that yielded the highest prediction accuracies (MAC, BAC, and overall-AUC) for prediction of student academic performance for INT. It should be noted the actual values of overall-AUC computed via R were converted into multiples of 100 for visual acuity in figures 6.10 and 6.11.

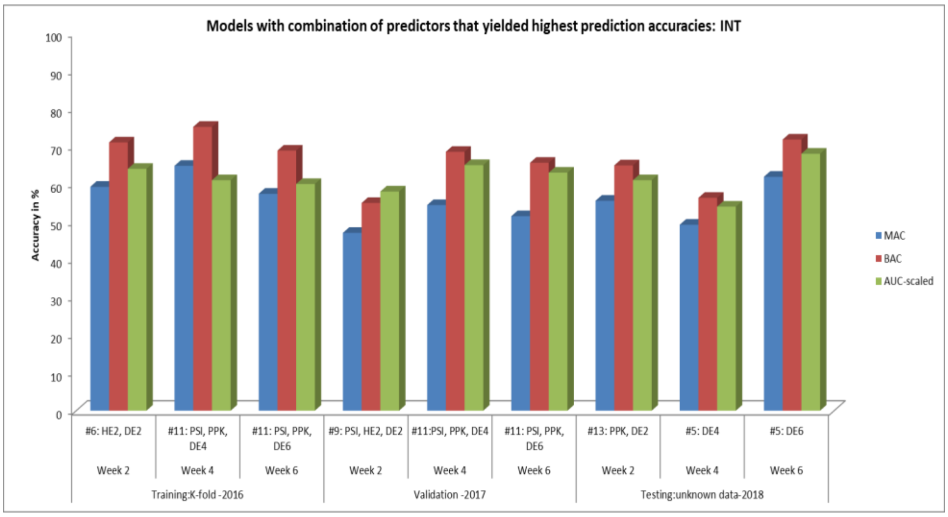

Figure 6.10: The overall prediction accuracies on training (K-fold), validation and testing on Week 2, Week 4 and Week 6 for INT.

The K-fold cross-validation results for Week 2 data (2016) revealed that the model developed with HE2 and DE2 as predictors for INT, had nearly moderate prediction accuracy (MAC: 59%) at identifying student FEG. In addition, this model is capable of correctly predicting the probabilities for students as being at risk of failing for 71% of the students (AUC: 0.64). The unknown data test results for Week 2 (2018) revealed that the model with PPK and DE2 as predictor variables returns moderate prediction accuracy (MAC: 56%) at identifying student FEG in compliance with AUC score 0.61. However, the validation results for Week 2 revealed that the model with HE2, DE2 and PSI on Week 2 (2017) yielded poor prediction accuracy (MAC: 47%) on identifying student FEG (AUC: 0.58).

The K-fold and validation results for Week 4 data revealed that the model with PSI, PPK and DE4 returns the good (MAC: 65%) and moderate (MAC: 54%) prediction accuracies at identifying student FEG in the years 2016 (K-fold cross-validation) and 2017 (validation), respectively. However, the unknown data test results for Week 4 revealed that the model with DE4 only as predictor, identified as the best predictor and had nearly moderate prediction accuracy (MAC: 49%) at identifying student FEG in compliance with AUC score (0.54). The K-fold (2016) and validation (2017) results for Week 6 revealed that model with PSI, PPK and DE6 as predictors returns best combinations of predictors that yielded moderate predictive accuracies at identifying FEG in INT. On the other hand, unknown data test results for Week 6 revealed that the model with DE6 only as predictor had good prediction accuracies at identifying student FEG in compliance with AUC score (0.68). As noted in section 6.1.1, the models developed with DE or combination of cognitive variables PSI or PPK or both yielded the highest prediction accuracies at identifying student FEG in INT.

Similarly, Figure 6.11 shows the models with different predictor combinations with different data sets (Week 2, Week 3, and Week 4) yielded the highest prediction accuracies (MAC, BAC, and overall-AUC) for prediction of student's academic performance in ALG.

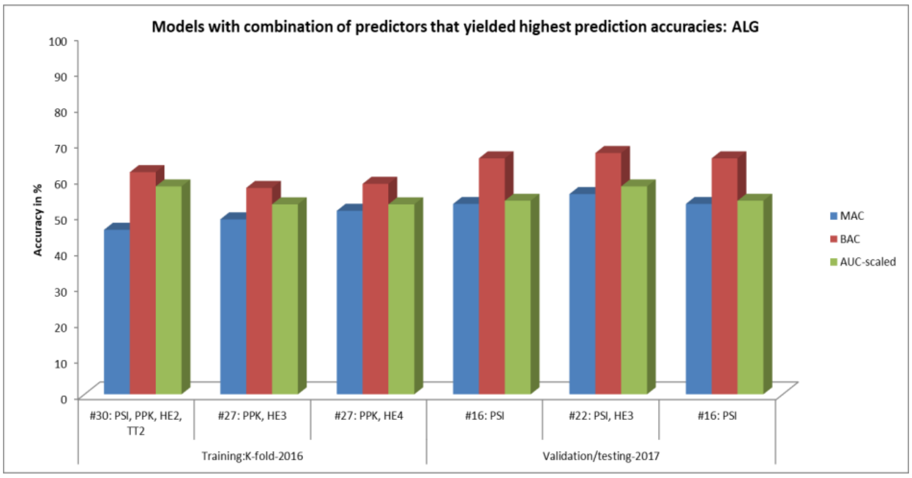

Figure 6.11: The overall prediction accuracies on training (K-fold) and testing on Week 2, Week 3 and Week 4 for ALG.

The K-fold cross-validation results for Week 2 and Week 3 (2016) had poor prediction accuracies (MAC: 46% and MAC: 49%) at identifying FEG, although K-fold cross-validation results for Week 4 yielded moderate prediction accuracy (MAC: 51%) for identification of FEG (AUC: 0.53). In addition, these results revealed that models with PPK and HE only as predictors for Week 3 and Week 4 were capable of correctly predicting students as being at risk of failing for 58% (on average: BAC) of the students in compliance with AUC score 0.53, compare to models with other combinations of predictors.

The unknown data test results for Week 2, Week 3, and Week 4 for ALG (2017) yielded mixed results (Figure 6.11). The model with PSI and HE3 (#22) as predictors yielded moderate prediction accuracy (MAC: 56%) at identifying FEG for Week 3 and in compliance with AUC score 0.58. However, the unknown data test results for Week 2 and Week 4 revealed that none of the models with different combinations of predictors yielded higher prediction accuracies with low false positives, over models with PSI only as predictors. On the other hand, these identified models had highest prediction accuracies, on the probability of identifying not at-risk students (NPV in between 78% and 82%) with high false negatives raised the predictive capabilities of these models. Furthermore, models developed and tested with various combinations of selected variables for ALG results, revealed that with cognitive variables PSI, and combinations of formative assessments TT or HE or both, the models yielded moderate prediction accuracies on predicting student FEGs in compliance with AUC scores (between 0.53 – 0.58).

6.3 Identifying academically at-risk students

One key objectives of this study was to identify at-risk students that need support, early, in order to alleviate their learning difficulties. As such, the at-risk prediction accuracy was measured based on number of students who received the final exam grade "0" (fail) or "1" (marginal pass). For example, if the student's FE score was less than 50 in *Introduction to Programming* or less than 45 in *Algorithms and Programming*, then his/her grade will be "ZERO" and it will be denoted as "0" in the student's transcript of study records of respective courses. Hence, this study tags students FEG who secured grade 0 (fail) or 1 (marginal pass) as at-risk students, to check the at-risk student prediction accuracy of the model (Tables 5.4 and 5.6). We calculated the ATSE, ATSP, and AUC score for at-risk class versus all other not-at-risk classes based on the test set of final exam grades, computed across 10-trials of cross-validation with actual final exam grades. Then, these developed models were validated, and tested by using unknown data for generalisation. Tables 6.1 and 6.2 present models that had highest at-risk prediction accuracies with low false positive rates in compliance with AUC score for at-risk class versus all other classes (binary classification) based on Week 2, Week 4 and Week 6 data for INT in the years 2016, 2017 and 2018 and for ALG in the years 2016 and 2017.

Dataset and year	**Week**	**No.**	**ATSE**	**ATSP**	**AUC: at-risk Vs all other classes**	**95% CI**
K-fold cross-validation: 2016	Two	#6	57.14	84.85	0.71	0.59-0.84
	Four	#11	71.43	78.79	0.75	0.63-0.87
	Six	#11	61.9	75.76	0.69	0.56-0.82
Validation: 2017	Two	#2	81.25	44.23	0.63	0.51-0.75
	Four	#11	81.25	55.77	0.69	0.57-0.81
	Six	#11	82.25	50.00	0.66	0.54-0.78
Unknown data testing: 2018	Two	#13	82.76	47.06	0.65	0.54-0.76
	Four	#5	86.21	26.47	0.56	0.47-0.69
	Six	#5	75.86	67.65	0.72	0.61-0.83

Table 6.1: Models had highest at-risk prediction accuracies with 95% CI for AUC: INT

The statistical results for model development, validation, and unknown data for INT produced good results (Table 6.1). On average, the ATSE for identifying students that need support for INT was 63% (2016), 82% (2017) and 82% (2018) in compliance with moderate and good ATSP and AUC scores for at-risk class versus all other classes (between 0.56-0.75). However, the unknown data test results on Week 4 produced high sensitivity (ATSE: 86%) with low specificity (27%) at identifying at-risk and not at-risk students in compliance with AUC score 0.56.

The statistical results for training and unknown data testing for ALG produced mixed results (Table 6.2). For example, K-fold cross-validation results for Week 2, Week 3 and Week 4 data discovered that models developed with all selected variables as predictors identified as model that had highest prediction accuracy but poor prediction accuracies on identifying at-risk students but with high ATSP or high false positives in compliance with AUC scores > 0.5. The unknown data test results on models with different combination of predictors did not yield any significant/good prediction accuracies on identifying at-risk prediction accuracies in compliance with AUC scores, although model #22 for Week 3 yielded moderate prediction accuracies. However, the model with PSI only as predictor had good prediction accuracies on identifying both at-risk and not at-risk students in compliance with AUC score 0.66 despite the student formative assessment scores secured in Week 2, Week 3 and Week 4. In addition, most of the models with different combinations of predictors yielded moderate prediction accuracies had insignificant CI for AUC scores (Appendix 9.5).

Dataset and year	Week	No.	ATSE	ATSP	AUC: at-risk Vs other classes	95% CI At-risk
K-fold cross-validation: 2016	Two	#30	47.73	76.13	0.62	0.54-0.70
	Three	#30	36.36	81.75	0.55	0.51-0.67
	Four	#30	36.36	84.13	0.6	0.52-0.68
Unknown data testing: 2017	Two	#16	60.71	70.94	0.66	0.56-0.76
	Three	#22	50.00	84.62	0.67	0.56-0.77
	Four	#16	60.71	70.94	0.66	0.56-0.76

Table 6.2: Models had highest at-risk prediction accuracies with 95% CI for AUC: ALG

Although the aforementioned results confirm that it is possible to predict student performance and identify at-risk student early in the semester, they are unable to identify a single model with a suitable feature subset that can be proposed as early warning systems. As such, the ensemble method was deployed to improve overall at-risk prediction performance: the results, reflecting early warning signs, may be incorporated in early warning systems. Consequently, an ensemble model was deployed to combine multiple predictions generated by models that yielded highest prediction accuracies in identifying at-risk students to get final predictions and propose those results as early warning signs. As such, models in the year 2018 for INT were selected based on 2016 and 2017 results. That is, DE, and its combination with other variables HE, PPK and PSI had higher predictions accuracies in the years 2016 and 2017 for INT (Figures 6.1 and 6.2). As such, at-risk prediction results of models with DE, and its combination with other predictors (#5-#6, #8-#9, #11, #13-#15) of 2018 were chosen to combine at-risk predictions for INT. However, the K-fold cross-validation results of 2016 for ALG made us to surmise that it is difficult to identify the best combination of predictors that yield significant prediction accuracies in ALG due to its assessment policy on final exam. As such, we did not deploy ensemble method for ALG.

6.4 Ensemble of classifiers

As noted, for this study, 15 x 2 models were developed, validated and tested to get final predictions. The models were selected based on training and validation results for ensembling via a majority voting method. Majority voting is a process of taking prediction with maximum votes (>=50%) from the multiple model predictions while predicting the outcomes of a classification problem. Figure 6.12 shows the at-risk prediction results of ensembling at-risk classifiers computed for unknown dataset after Week 2, Week 4, and Week 6 for INT.

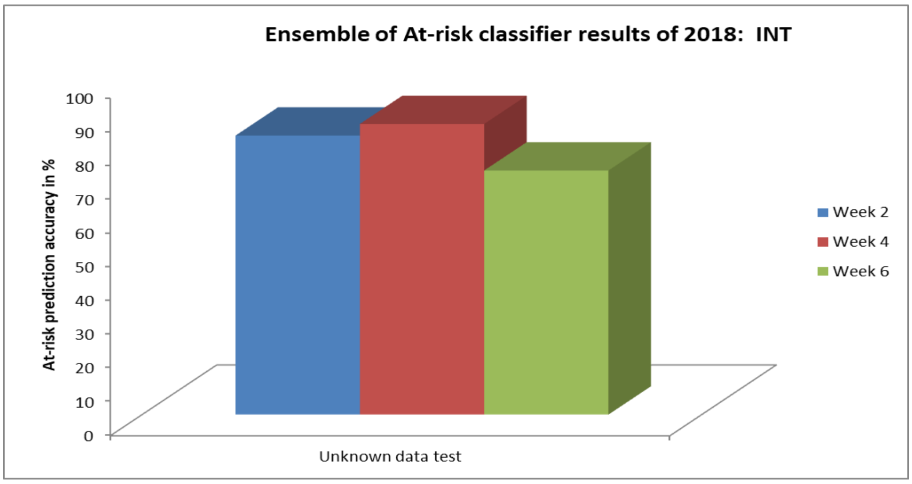

Figure 6.12: Ensemble of at-risk classifier results in the year 2018 for INT

The ensemble of at-risk classifiers results on unknown data test for INT show that on average, it is possible to identify 83% of students after Week 2, 86% of students after Week 4 and 72% of students after Week 6 that need support (Figure 6.12).

6.5 Summary of results

A total of 15 models x 2 models have been developed, validated, and tested by using Naïve Bayes classification technique. The features included: PSI, PPK, HE, and DE or TT. The K-fold cross-validation on test set, validation, and unknown data test results on models with a single feature as predictor revealed that, of the four features, DE is the most influential feature in predicting student FEG in INT with overall average AUC 0.59. The next average AUC (0.58) was provided by PSI and PPK (AUC 0.57). Similarly, PSI and followed by TT were identified as most predictive (both K-fold and testing) for ALG with moderate AUC scores (0.56 and 0.52). HE and PPK were identified as least influential features on predicting student performance in INT and ALG respectively although the predictive performance of models with HE and combination other features had nearly moderate or good prediction accuracies in both courses. The models with cognitive features (PSI & PPK), only, as predictors return slightly higher overall AUC (0.59) than models with formative assessments (HE and DE), only, as predictors in model development, validation and unknown data testing for Week 2, Week 4 Week 6 data (AUC: 0.58) for INT. However, the models developed and tested with formative assessments HE and TT, only, as predictors for ALG had poor/insignificant prediction accuracies. Moreover, the models developed and tested with different combination of predictors for ALG also had imbalanced prediction accuracies on identifying student FEG and or at-risk students or with high FP/FN. The majority voting –ensemble method results show that it is possible to predict students that need support within first Six weeks of the course period although there is no consistency in prediction accuracies between the results for Week 2, Week 4 and Week 6 for INT (Figure 6.12). However, these aforementioned statistical results imply that it is possible to visualise at-risk results obtained via ensemble modelling as early warning signals.

Chapter 7

Utilising predictive models as early warning systems: Discussion and conclusions

This chapter summarises the major research finding of our publications and replication and extended study conducted for this thesis, deployment of models as early warning systems, limitations and possible future work of this study. In Section 7.1 we present the contributions of publications to research questions and in section 7.2 we discuss the outcomes or findings in the context of the research questions (RQs) we set out to answer. Section 7.3 presents our conclusions and further work options.

7.1 Contributions of publications to research questions

Publication & Description of the study	Key findings	Contributions to research questions
P1. This study examined the influence of lecture attendance on continuous summative assessment tasks and the subsequent final examination.	-Attending formal lecture sessions has no impact on student final exam performance. -Continuous summative assessments have impact on final examination. However, the strength of the relationship between the selected assessment tasks and the final exam performance varies from one academic year to next. -These results provide immediate information for novice programming course instructors to analyze further to find the factors that prevent novices from attending programming formal lecture sessions.	RQ1: Which feature selection techniques should be used to identify the influential factors that affect student learning and academic progress based on available academic data? ➢ How to identify the factors that foster students' learning performance in computer programming courses? ➢ How to determine the course specific factors that influence students' academic performance?
P2.This study is an extension of our prior study P1 in which we raised the question "why some	-Prior programming knowledge has a significant impact on student lecture attendance.	

Publication & Description of the study	Key findings	Contributions to research questions
students skip lecture sessions yet, do well in the final exam?" This study explored the impact of prior knowledge on lecture attendance and on subsequent final examination in introductory programming course.	-Student with no prior knowledge attended lecture sessions more regularly than those with some prior programming knowledge. -There was no significant difference in the distribution of lecture attendance between students with basic and higher levels of prior programming knowledge. -There is a statistically significant difference in final exam scores between the students with no prior programming knowledge and those that with some prior programming knowledge. -Prior programming knowledge affects student academic achievement in programming courses. -Lecture attendance has no impact on student final examination performance. -PPK can be used to determine student progress and performance.	
P3. This study was conducted to examine the relevance of problem-solving skills in student performance in ongoing assessment tasks and final programming exam.	-There was no statistical significant difference in ongoing assessment task scores between the students with different problem-solving skills. -Problem solving skills has a significant impact on student final exam performance. -There is a difference in final exam scores between students with good versus those with poor problem-solving skills.	

Publication & Description of the study	Key findings	Contributions to research questions
	-It is possible to categorize students on the basis of PSS, to explore student constructivist learning improvements. -Measuring student PSS in the beginning of novice programming course can be useful in predicting the student final programming exam performance in the course.	
P4. The objective of the research reported in this study was to develop a predictive model with selected predictor variables using support vector machine algorithm to predict student performance and at-risk students in a programming course (at university level) to make proactive measures in teaching and learning. This study attempted to explore the impact of formative assessment tasks and prior programming knowledge in predicting student's final exam scores.	-The success rate of the model is 52% on predicting student final exam scores of all students in the programming course. -The statistical results of binomial test revealed that the model has a 46% success rate for predicting academically at-risk students and not significant. -The comparison between MSE/RMSE values of training and validation sets suggest that the model is slightly over fitted. -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. -The study results suggest that develop a simple model(s) with explanatory predictor variables, with selection based on the principle of parsimony and previous research findings.	RQ2: How can a predictive model be developed and validated to predict performance in programming courses? RQ4: What percentage of academically at-risk students may be correctly identified by the model? ➢ How to develop and validate a mathematical model using the educational data collected from programming courses? ➢ How to predict student final exam scores using the collected educational data via learning management systems? ➢ How to measure the predictive accuracy of regression model? ➢ How to identify student that need support from predicted values in a developed model? ➢ Which machine learning algorithm to be used to construct

Publication & Description of the study	Key findings	Contributions to research questions
		predictive models for student performance?
P5. The main objective of this study was to construct a predictive model with a combination of predictor variables that predict final programming exam performance of students.	-The models developed with single predictors for introductory programming courses revealed that, models developed with DE or TT (followed by PSI and PPK) as the most influential factor in determining student final exam performance in compare to other features used in this study. -The at-risk student prediction accuracy on k-fold test result is good and reveals that it is possible to predict 81% (average of top Three models' at-risk prediction accuracies) of students who need early assistance in introductory programming courses, based on their problem-solving skills, and scores achieved in selected formative assessment tasks, in the first few weeks of the semester. -Hence, these results imply that our model may be adapted as an EWS in programming courses that has continuous assessment and final exam components, to predict student academic performance and to identify students that need support. -The model(s) may be used by instructors to categorize students as, for example, "at-risk", "marginal", "average", "good", "very	RQ3: What combination of predictor/independent variables yields the highest prediction accuracy to predict student's academic performance? RQ4: What percentage of academically at-risk students may be correctly identified by the models? RQ5: How suitable are developed models for incorporation in early warning systems, for educators to identify students that need assistance in introductory programming courses? ➢ 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 at early stage of the course? ➢ Might our proposed model with these predictor variables be deployed as EWS to support instructors and programming students?

Publication & Description of the study	Key findings	Contributions to research questions
	good", and "excellent" based on predicted final exam grades, in order to reshape their pedagogical practices accordingly. -Based on the past research findings and results of this study, a generic predictive model was proposed that can be deployed for other programming and non-programming courses, if the goal of instructor is to predict student performance early in the semester.	➢ Might our proposed model be transformed as a generic predictive model for other courses that has continuous summative assessments and or final exam, to predict student performance early in the semester?
Replication-extension study. This study was conducted to verify P5 using larger dataset with different structure to to know how well the models we developed and validated in prior studies will perform on future or on unknown data. Sections 5-7 present the replication study in detail.	-The replication study results revealed classroom assisted formative assessments influence student performance in programming exam. -The majority voting – ensemble method results suggest that it is possible to predict student performance and identify students that need support from second week of the semester onwards.	

Table 7.1: Contributions of publications to research questions.

7.2 Discussion of the results: Answers to research questions

RQ1. *Which feature selection methods should be used to identify the influential factors that affect student learning and academic progress based on available academic data?*

Identifying influential factors that contribute to student learning and academic progress is important in machine learning as it helps in understanding data, reducing computation requirement, and better model interoperability. Moreover, including unnecessary features in a model will influence the predictive performance of a model. We used filter methods for P1, P2, and P3 and wrapper methods for P4, P5, and replication-extension study to identify potential factors that influence student performance in programming. First, the predictor variables were selected based on their intrinsic properties measured via correlation coefficient and linear regression techniques (filter method). Then these selected factors were evaluated again in a model by using various machine learning algorithms with cross-validation techniques and different selection procedures (wrapper method) such as forward selection or stepwise regression to find optimal features based on the learning performance.

In the filter method all features of the dataset ranked based on certain criteria (correlation for example) to lets the researcher to select the features those with highest rankings are as predictors before the deployment of machine learning algorithms. As such, we used filter method (Spearman's Rank correlation and linear regression techniques) in P1 to examine the relationship and influence of LEA, HE and DE on student performance in final programming exam to determine if LEA and FA might be suitable predictors of student performance (Veerasamy, et al., 2016). We identified that LEA and student performance in FE was negatively correlated and statistically insignificant. On the other hand, student FAs scores and performance in FE was positively correlated and statistically significant. However, correlation does not imply causation. As such, the multiple linear regression technique was deployed to measure the impact of LEA and FA on FE scores. This is because, multiple linear regression examines how an independent variable is numerically related to the dependent variable and the results of multiple regression indicates the impact of a change in value of independent variable on the value of dependent variable. The multiple linear regression results of P1 revealed that the effect of HE and DE (FAs) on FE is moderate and significant respectively. However, the effect of LEA on the FE is not significant. As such, our subsequent studies (P4, P5 and replication study) included HE and DE as features for model development and excluded LEA for further analysis. Similarly, we conducted two more studies (P2 and P3) to identify influential factors that affect student learning. The P2 and P3 results revealed that PPK and PSS (cognitive factors) influence student performance in programming courses (Veerasamy, Daryl D'Souza, & Laakso, 2018; Veerasamy;D'Souza;Lindén;& Laakso, 2018). We used filter method in P4 to select potential factors for Support vector machine regression-based predictive model development (Veerasamy;D'Souza;Lindén;& Laakso, 2019). The multiple linear regression results of P4 (Adjusted R square: 0.264) revealed that there is a relationship between PPK, HE and DE on FE performance and can be used as predictors for model development. However, although the overall prediction accuracy of the model is moderate, the prediction accuracy on identifying at-risk students was not significant. It should be noted, filter method is simple and computationally inexpensive. However, filter method determines the features that have higher variance and filter out the least promising features. But it ignores feature dependencies which may lead to poor classification performance. So, it might have failed to find the best subset of features. On the other hand, wrapper feature selection method is model oriented and gets good subset of features using the machine learning algorithm itself as part of the evaluation function (Li, et al., 2017). As such, for P5 we used wrapper method for feature selection to identify the best subset of features that could predict student performance. In addition, this method was used to examine the features that had highest predictive performance for model development and validation for RQ3. The feature selection results (evaluated by machine learning) of P5 shown that student PSI, PPK, HE, DE/TT had moderate or good influence on predicting student FEG in programming courses.

However, our replication study results conducted on INT course revealed that, DE is the most influential factor that influence student performance and followed PSI, PPK and HE (Figures 6.1, 6.2 and 6.3). On the other hand, the wrapper method results conducted on ALG course revealed that, PSI and followed by TT, were identified as most influential factors that affect student performance (Figures 6.4 and 6.5).

RQ2. *How can a predictive model be developed and validated to predict performance in programming courses?*

One of the objectives of this research was to develop a model using course specific academic and cognitive variables extracted from LMS (ViLLE) to predict student performance and identify students that need support to make proactive measures in teaching and learning. This can be achieved by utilising predictive analytics techniques called predictive modeling. As known, the objective of predictive modelling in education is to predict student performance. So, the regression-based predictive model was initially deployed in P4 to answer our RQ2 (Veerasamy;D'Souza;Lindén;& Laakso, 2019). We used filter method in P4 for feature selection and developed a model using support vector machine (SVM) learning-regression algorithm for prediction of student final programming exam scores. SVM-regression is generally good for numerical prediction and has a high generalisation performance. K-fold cross-validation was used to estimate the performance of a model to know how well the developed model will work on unknown data. Although the overall prediction accuracy of the model was moderate, predictions on identifying at-risk students was neither high nor low. Hence, the results of P4 persuaded us to update research methodology to improve predictive performance For example, inclusion of more data, and features, check for multicollinearity symptoms if exists between the input variables, and exploring the impact of other machine learning algorithms as some algorithms might work well better on certain types of dataset than others. In addition, P4 results made us to conclude that identifying students that need support early in the semester would assist instructors to take necessary interventions.

Therefore, classification algorithm based predictive models would serve better than regression models that we tested in P4. As such, we conducted P5 to construct Naïve Bayes classification-based models (selected based on preliminary prediction performance results over other algorithms such as Random forest, SVM, C5.0) to answer the research questions RQ2 and RQ3. As noted in P4, including too many variables that provide similar information will bring the issue of multicollinearity and may affect model's predictive performance (Veerasamy;D'Souza;Lindén;& Laakso, 2019). As such, we deployed parsimonious modelling procedures in P5 to develop a predictive model with a minimum set of explanatory predictor variables selected based on prior studies (P1-P4) and tested on unknown data for generalisation. The P5 results (both K-fold cross-validation and testing) revealed that student PSI, PPK, HE and DE/TT captured in predictive model was a good fit of the data.

Our replication study on unknown data testing results revealed that, it is possible to predict student performance and identify students at-risk of course failure although the consistency between the predictors combinations and results of models varied year to year (Tables 6.1 and 6.2 and Figures 6.10 and 6.11).

RQ3. *What combination of predictor/independent variables yields the highest prediction accuracy to predict student's academic performance?*

To answer RQ3 we developed Naïve Bayes classification (NBC) based models in P5. We used wrapper method to find suitable subset of features as it was learned from P4 that selection of features and type of machine learning algorithms play vital role in predictive model development and performance. As such, NBC was used for predictive model development in P5 as they provided better prediction accuracy on identifying at-risk students compared to other machine learning algorithms such as Support vector machine, Random forest and C5.0. Variance inflation factor analysis was deployed to assure that models developed for P5 had no highly correlated predictor variables that can cause multicollinearity. The validation results of P5 shown that the models developed with DE and or combination of cognitive features PSI and PPK yielded highest prediction accuracies on predicting student performance in a programming course. The P5 results also suggest that parsimonious models are likely to perform better on unknown or test data than models with many predictor variables.

Our replication-extension study on unknown data testing yielded mixed results on identifying the suitable combination of predictors that yielded the highest prediction accuracy on prediction of student performance. As noted earlier, the models tested with predictors DE, and or combination of PSI and PPK on unknown data (2018) had highest prediction accuracies on predicting student performance in INT (Table 6.1). However, the models developed and tested for ALG course produced insignificant results although the models with PSI or PPK or both had nearly moderate or moderate performance on predicting student FEG (Table 6.2). These results made us to surmise that models with DE and combination of other predictors might yield better prediction accuracies than other models for INT and answered RQ3. However, the predictors selected for model development in ALG need to be tuned as current models with selected features did not yield significant results as expected and it should be analysed further (Figure 6.11).

RQ4. *What percentage of academically at-risk students may be correctly identified by the model?*

The motivation of this research was high failure/attrition in introductory programming courses affect both time-to-graduation and student retention. So, this research was focussed on developing models for identification of students that need support early in the course for instructors to provide timely aid to those students. Two different studies (P4 and P5) were conducted to answer RQ4 of this research. As noted earlier, we deployed regression based predictive modelling in P4 and classification based predictive modelling in P5 to identify at-risk students in programming. Regression based models are mainly used where the researcher target is to predict continuous quantity such as marks, income for example. On the other hand, classification-based models are mainly useful for predicting a label of an observation (For example, pass or fail, excellent or good or poor).

We developed a model with three factors PPK, HE and DE as input for P4 to predict student FE scores (regression). The binomial test result on probability of identifying at-risk students was nearly moderate (0.462). It was identified that the selection of features influenced model's prediction accuracy suggesting that, including one or more predictor variables in the model may improve the model's accuracy on predicting student performance and identifying at-risk students. Furthermore, we used support vector machine algorithm to train and validate the model using K-fold cross-validation. The results of P4 suggested that, this study could be replicated by using various other machine learning algorithms to check the performance of the model for identifying at-risk students in programming as the learning process of machine learning algorithm can be influenced by the dataset used for training and testing and in turn it might influence model's performance. In addition, the objective of the study was to classify students that need support in order to provide necessary teaching interventions to those identified group of students. Therefore, we used NBC algorithm for model development and validation in P5 to identify answers for RQ4 and RQ5. The PSI was included with other factors PPK, HE and DE in model development for identification of student-at-risk in programming. Two (courses) X 15 predictive models were developed to identify the models with predictors that yielded highest prediction accuracies on identifying student at-risk in introductory programming. On average, the prediction accuracy in identifying at-risk students for introductory programming courses on the test set was 71% and 59%. The statistical results on identifying at-risk students of P5 imply that it is possible to identify at-risk students in the first four weeks, based on student PSI, PPK, HE and DE in introductory programming course. The results of P5 motivated us to conduct another study with more data with different structure to verify the results of P5.

We replicated P5 work with more data and models developed with same set of features. The models were validated, and tested after Week 2, Week 4 and Week 6 for INT and after Week 2, Week 3 and Week 4 for ALG course to check how well these models tested in P5 works on unknown data with different structure. The replication and extended study results on unknown data for INT shown that, it is possible to predict student that need support after Week 2 (83%), Week 4 (86%) and Week 6 (76%) (Table 6.1). Similarly, the unknown data test results on ALG revealed that it is possible to predict 61% of students that need academic support based on PSI early in the course (Table 6.2). In addition, our ensemble method results of unknown data for INT confirm that on average, it is possible to identify 83% of students after Week 2, 86% of students after Week 4 and 72% of students after Week 6 that need support with low false positives. However, we did not deploy ensemble method for ALG to obtain improved at-risk prediction accuracies as most of the models with different combinations of predictors did not yield significant predictions.

RQ5. *How suitable are developed models for incorporation in an early warning system for educators to identify student that need assistance in introductory programming courses?*

As known, developing and employing an early warning system that tracks student progress through the analysis of readily available student academic and cognitive data is critical for higher education to identify students that need support and to refer them relevant support activities before they reach critical point. As such, we developed and validated set of predictive models in P5 that can be proposed as early warning systems for programming courses. The statistical results of P5 revealed that the models developed and tested in this study can be adopted as early warning systems. These models can be very useful to track the progress of individual students after week 4. In addition, based on the research finding and results of P5 a generic predictive model was proposed, which can be deployed for other programming and non-programming courses for instructors to predict student performance early in the semester.

However as noted, it is important identify student that need support as early as possible in order to understand the root causes of student engagement and academic failure to provide more effective intervention services. So, we conducted a replication and extended study based on P5 results with the objective of developing predictive models that capable of identifying at-risk students from beginning of the semester (Week 2 onwards) in order to incorporate those models as early warning systems. The statistical results on unknown data test of our replication study revealed that (Table 6.2) it is possible to identify 61% of students that need support in the beginning of the semester (Week 2) based on student PSI survey responses in ALG. Similarly, 83% of students can be identified in INT based on student performance in DE with other cognitive variable PPK after Week 2 (Table 6.1). In addition, the ensemble of at-risk classifiers results on unknown data test for INT that it is possible to identify students that need support in the early weeks (after Week 2) of the semester (83%). As such, the models developed for this study can be incorporated as an early warning system to identify students that need support after Week 2, Week 4 and Week 6 for INT course (Figure 6.12).

7.3 Conclusions

Identification of students that need support in programming has been a long-standing problem. In this thesis, we developed a set of validated parsimonious predictive models to predict student academic performance in introductory programming courses to identify at-risk students early in the semester, by using presage (cognitive variables) and in-progress factors (formative assessments) as predictor variables. 15 x 2 models were developed, validated and tested by using different data sets collected during the Week 1 to Week 6 periods of the semesters. Model prediction sensitivity, specificity, and positive and negative predicted values, and balanced accuracies, were computed via a confusion matrix to weigh and compare the prediction accuracy of the models. The influence of features evaluated by using stepwise regression techniques identified that DE, PSI, and PPK were the most valuable factors influencing the predictive performance of the models. The statistical results of unknown data tests showed that overall success of the models was moderate and good and that these models may therefore be incorporated in early warning systems to assist instructors to identify students that need early assistance. In addition, unknown data test results suggest that instructors may use PSI and PPK responses (from students) as predictors to identify students that need support before they engage in course assessment tasks.

Additionally, as our models were developed by using a multiclass classification-based algorithm, the models may be used by instructors to categorise students as “at-risk and marginal pass”, “good” and “very good”, based on predicted final exam grades, to reshape their pedagogical practices, accordingly. Similarly, it is possible to understand the student PSI and PPK levels early in the course to develop inclusive teaching strategies, to engage students with varied programming knowledge and problem-solving skills. As noted in P3, it is possible to categorise students on the basis of PSS to explore student constructivist learning improvements (Veerasamy;D'Souza;Lindén;& Laakso, 2018). For example, providing course assessment tasks to promote student programing problem-solving skills and connect programming thinking. Similarly, as noted, the predictive models developed in this study were based on the data collected via ViLLE. So, it is quite possible to present these models results as early warning signals at ViLLE in visual form for instructors to identify students that need support early in the semester. Therefore, our publications and replication-extension study results provide the evidence that by analysing readily available student formative assessment data and course related cognitive data it is possible to implement effective interventions in order to avoid or minimise student failures.

7.4 Limitations and future work of the study

Although the results (ensemble) were good, this study has a number of limitations that influence the overall generalisability and internal validity of the proposed study. First, only a few cognitive features including PPK and PSI were concerned in this study. Second, this study used self-reported survey data to examine student PSI and PPK levels that may contain potential sources of bias; it is unknown whether or not students responded to the questionnaires honestly although Cronbach's Alpha, a psychometric test, on PSI and PPK reliability, yielded good values. Third, we used the first six weeks of assessment results for analysis. However, learning is dynamic and a learner might not do well in the first few weeks of the semester and may perform well in subsequent weeks of the semester. Hence, there is a need to monitor and track student progress throughout the course period in order to provide continuous academic support. Fourth, the findings presented in this research cannot be generalised as the data used in this study was collected within one institution although the models of this study can be tested to other programming courses. Fifth, although predictor variables used in this study yielded moderate and good results, there still remains a degree of uncertainty as to which variables or combination of variables has the most predictive power.

This study may be extended to develop ensemble models of various machine learning algorithms by using similar set of features and or other predictor variables that could influence the performance of students for multiple courses across a curriculum and at multiple institutions. Based on the past research findings and results of our replication and extended study our predictive model(s) can be deployed for other programming and non-programming courses, if the goal of instructor is to predict student performance early in the semester. This study can be extended like "how to use our previously developed predictive models as early warning systems, to identify students that need early attention/support to alleviate any potential for becoming at risk" In addition, this study can be extended to to investigate the effectiveness of a visualization tool to serve as an early warning system (EWS) for introductory programming courses.

8. References

Adamopoulos, F. A. (2017). *An Influence Model of the Experience of Learning Programming.* RMIT University, School of Business IT and Logistics. Melbourne: RMIT University.

Aguiran, J. M., Lazo, R. B., & Salabat, M. J. (2014). Factors affecting students' performance in research. *CVCITC Research Journal, 1*(2), 37-47.

Ahadi, A., Lister, R., Haapala, H., & Vihavainen, A. (2015). Exploring Machine Learning Methods to Automatically Identify Students in Need of Assistance. *Proceedings of the eleventh annual International Conference on International Computing Education Research* (pp. 121-130). Omaha, Nebraska, USA: ACM. doi:10.1145/2787622.2787717

Akar, S. G., & Altun, A. (2017). Individual Differences in Learning Computer Programming: A Social Cognitive Approach. *Contemporary Educational Technology, 8*(3), 195-213.

Alexander, V., & Hicks, R. E. (2016). Does Class Attendance Predict Academic Performance in First Year Psychology Tutorials? *International Journal of Psychological Studies, 8*(1), 28-32. doi:10.5539/ijps.v8n1p28

Alexandron, G., Armoni, M., Gordon, M., & Harel, D. (2012). The effect of Previous Programming Experience on the Learning of Scenario-Based Programming. *Proceedings of the 12th Koli Calling International Conference on Computing Education Research* (pp. 151-159). ACM.

Al-Shehri, H., Al-Qarni, A., Al-Saati, L., Batoaq, A., Badukhen, H., Alrashed, S., . . . Olatunji, S. O. (2017). Student performance prediction using Support Vector Machine and K-Nearest Neighbor. *EEE 30th Canadian Conference on Electrical and Computer Engineering* (pp. 1-4). IEEE. doi:10.1109/CCECE.2017.7946847

Anderson, H., Boodhwani, A., & Baker, R. (2019). Predicting Graduation at a Public R1 University. *Companion Proceeding of the 9th International Conference on Learning Analytics & Knowledge* (pp. 242-244). Tempe: ACM.

Arnold, K. E., & Pistilli, M. D. (2012). Course signals at Purdue: using learning analytics to increase student success. *Proceedings of the 2nd International Conference on Learning Analytics and Knowledge* (pp. 267-270). Vancouver, British Columbia, Canada: ACM. doi:10.1145/2330601.2330666

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), 49-61. doi:10.5815/ijisa.2015.01.05

Astin, A. W. (1978). *Four Critical Years. Effects of College on Beliefs, Attitudes, and Knowledge.* ERIC.

Atif, A., Richards, D., & Bilgin, A. (2015). Student Preferences and Attitudes to the Use of Early Alerts. *Twenty-first Americas Conference on Information Systems*, (pp. 1-14). Puerto Rico.

Austin, P. C., & Steyerberg, E. W. (2015). The number of subjects per variable required in linear regression analyses. *Journal of Clinical Epidemiology*, 627-636. Retrieved from https://doi.org/10.1016/j.jclinepi.2014.12.014

Austin, P. C., & Tu, J. V. (2004). Bootstrap Methods for Developing Predictive Models. *The American Statistician, 58*(2), 131-137. doi:10.1198/0003130043277

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

Baepler, P., & Murdoch, C. J. (2010). Academic Analytics and Data Mining in Higher Education. *International Journal for the Scholarship of Teaching and Learning, 4*(2). doi:10.20429/ijsotl.2010.040217

Bai, X., Ole, A., & Akkaladevi, S. (2018). Examination of the relationship between class attendance and student academic performance. *Issues in Information Systems, 19*(3), 191-109.

Baker, R. S., & Inventado, P. S. (2014). Educational Data Mining and Learning Analytics. In J. A. Larusson, & B. White, *Learning Analytics – From Research to Practice* (pp. 61-74). Springer Publishing Company.

Bawamohiddin, A. B., & Razali, R. (2017). Problem-based Learning for Programming Education. *International Journal on Advanced Science, Engineering and Information Technology, 7*(6), 2035-2050.

Behjoo, B. M. (2013). *The Relationship Among Self-Efficacy, Academic Self-Efficacy, Problem Solving Skills and Foreign Language Achievement.* Master thesis, Hacettepe University Graduate School of Social Sciences, Department of Foreign Language Teaching , Ankara.

Bennedsen, J., & Caspersen, M. E. (2007). Failure Rates in Introductory Programming. *ACM SIGCSE Bulletin, 39*(2), 32-36.

Bennedsen, J., & Caspersen, M. E. (2019, Junel 25). Failure Rates in Introductory Programming — 12 Years Later. (M. Hamilton, & J. Harland, Eds.) *ACM inroads*, *10*(2), pp. 30-36. doi:https://doi.org/10.1145/3324888

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

Bester, L. (2014). *Investigating the problem-solving proficiency of second-year Quantitative Techniques students : the case of Walter Sisulu University*. PhD Dissertation, University of South Africa, Department of Education, Pretoria. Retrieved from http://hdl.handle.net/10500/14214

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, 54*, 2976-2989. doi:10.1016/j.csda.2010.03.004

Buskes, G., & Belski, I. (2017). Prior knowledge and student performance in idea generation. In N. Huda, D. Inglis, N. Tse, & G. Town (Ed.), *28th Annual Conference of the Australasian Association for Engineering Education (AAEE 2017)* (pp. 354-361). Sydney: Australasian Association for Engineering Education.

Bydžovská, H. (2016). A Comparative Analysis of Techniques for Predicting Student Performance. In 2016 (Ed.), *Proceedings of the 9th International Conference on Educational Data Mining* (pp. 306-311). Raleigh, NC, USA: International Educational Data Mining Society.

Carter, A. S., Hundhausen, C. D., & Adesope, O. (2015). The Normalized Programming State Model: Predicting Student Performance in Computing Courses Based on Programming Behavior. *Proceedings of the eleventh annual International Conference on International Computing Education Research* (pp. 141-150). Omaha, Nebraska, USA: ACM. doi:10.1145/2787622.2787710

Castro-Wunsch, K., Ahadi, A., & Petersen, A. (2017). Evaluating Neural Networks as a Method for Identifying Students in Need of Assistance. *In Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education* (pp. 111-116). Seattle: ACM. doi:http://dx.doi.org/10.1145/3017680.3017792

Chandrashekar, G., & Sahin, F. (2014). A survey on feature selection methods. *Computers and Electrical Engineering, 40,* 16–28. Retrieved from https://doi.org/10.1016/j.compeleceng.2013.11.024

Chapin, L. A. (2018). Australian university students’ access to web-based lecture recordings and the relationship with lecture attendance and academic performance. *Australasian Journal of Educational Technology, 34*(5), 1-13.

Chatti, M., Dyckhoff, A., Schroeder, U., & Thus, H. (2012). A Reference Model for Learning Analytics. *International Journal of Technology Enhanced Learning, 4*(5-6), 318-331.

Chio, C., & Freeman, D. (2018). Why Machine learning and Security. In *Machine Learning and Security* (pp. 9-10). Sebastopol: O’Reilly Media, Inc.,.

Conijn, R., Snijders, C., & Kleingeld, A. (2017, January_March). Predicting Student Performance from LMS Data: A Comparison of 17 Blended Courses Using Moodle LMS. *IEEE Transactions on Learning Technologies, 10*(1), 17-29. doi:10.1109/TLT.2016.2616312

Costa, E. B., Fonseca, B., Santana, M. A., & Araújo, F. F. (2017, February 4). Evaluating the effectiveness of educational data mining techniques for early prediction of students' academic failure in introductory programming courses. *Computers in Human Behavior, 73,* 247–256. doi:https://doi.org/10.1016/j.chb.2017.01.047

Dekker, G. W., Pechenizkiy, M., & Vleeshouwers, J. M. (2009). Predicting Students Drop Out: A Case Study. *International Conference on Educational Data Mining*, (pp. 41-50). Cordoba, Spain.

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), 265-282. doi:10.1111/j.2044-8317.1992.tb00992.x

Devasia, T., P, V. T., & Hegde, V. (2016). Prediction of students performance using Educational Data Mining. *2016 International Conference on Data Mining and Advanced Computing (SAPIENCE)* (pp. 91-95). Ernakulam: IEEE.

Eren, O., & Henderson, D. J. (2008). The impact of homework on student achievement. (R. J. Smith, Ed.) *The Econometrics Journal, 11*(2), 326-348.

Erözkan, A. (2014). Analysis of Social Problem Solving and Social Self-Efficacy in Prospective Teachers. *Educational Sciences: Theory and Practice, 14*(2), 447-455. doi:10.12738/estp.2014.2.2014

Ettles, A., Luxton-Reilly, A., & Denny, P. (2018). Common logic errors made by novice programmers. *Proceedings of the 20th Australasian Computing Education* (pp. 83-89). Brisbane: ACM. doi:10.1145/3160489.3160493

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

Fan, H., Xu, J., Cai, Z., He, J., & Fan, X. (2017). Homework and students' achievement in math and science: A 30-year meta-analysis, 1986–2015. *Educational Research Review, 20*, 35-54. Retrieved from https://doi.org/10.1016/j.edurev.2016.11.003

Fawcett, T. (2006). An introduction to ROC analysis. *Pattern Recognition Letters, 35*(6), 861-874. doi:10.1016/j.patrec.2005.10.010

Feng, P.-M., Ding, H., Chen, W., & Lin, H. (2013). Naïve Bayes Classifier with Feature Selection to Identify Phage Virion Proteins. *Computational and Mathematical Methods in Medicine, 2013*, 1-6. Retrieved from http://dx.doi.org/10.1155/2013/530696

Francis, B. K., & Babu, S. S. (2019). Predicting Academic Performance of Students Using a Hybrid Data Mining Approach. *Journal of Medical Systems*, 1-15. Retrieved from https://doi.org/10.1007/s10916-019-1295-4

Gaal, F. V., & Ridder, A. D. (2013). The impact of assessment tasks on subsequent examination performance. *Active Learning in Higher Education, 14*(3), 213-225. doi:10.1177/1469787413498039

Gatsheni, B. N., & Katambwa, O. N. (2018). The Design of Predictive Model for the Academic Performance of Students at University Based on Machine Learning. *Journal of Electrical Engineering, 6*, 229-237. doi:10.17265/2328-2223/2018.04.006

Ghasemi, A., & Zahediasl, S. (2012). Normality Tests for Statistical Analysis: A Guide for Non-Statisticians. *International Journal of Endocrinology Metabolism, 10*(2), 486-489.

Gibbs, G. (2010). Pedagogic principles underlying the use of assessment to support learning. In G. Gibbs, *Using assessment to support student learning at University of East Angila* (pp. 11-23). Leeds Metropolitan University.

Gratchev, I., & Balasubramaniam, A. (2012). Developing assessment tasks to improve the performance of engineering students. *23rd Annual Conference of the Australasian Association for Engineering Education 2012*, (pp. 40-47). Melbourne.

Grover, S., Pea, R., & Cooper, S. (2016). Factors Influencing Computer Science Learning in Middle School. *Proceedings of the 47th ACM Technical Symposium on Computing Science Education* (pp. 552-557). Memphis, TN, USA: ACM.

Guo, B., Zhang, R., Xu, G., Shi, C., & Yang, L. (2015). Predicting Students Performance in Educational Data Mining. *2015 International Symposium on Educational Technology.* Wuhan, China.

Hailikari, T., Nevgi, A., & Komulainen, E. (2007). Academic self-beliefs and prior knowledge as predictors of student achievement in mathematics: a structural model. *Educational Psychology, 28*(1), 59-71. doi:10.1080/01443410701413753

Helal, S., Li, J., Liu, L., Ebrahimie, E., Dawson, S., & Murray, D. J. (2019). Identifying key factors of student academic performance by subgroup discovery. *International Journal of Data Science and Analytics, 7*, 227-245. Retrieved from https://doi.org/10.1007/s41060-018-0141-y

Heppner, P. P. (1982). The Problem Solving Inventory. New York: The American Psychological Association.

Hsu, W. C., & Plunkett, S. W. (2016). Attendance and Grades in Learning Programming Classes. *Proceedings of the Australasian Computer Science Week Multi conference.* Canberra.

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), 133–145.

Hussain, M., Zhu, W., Zhang, W., Abidi, S. M., & Ali, S. (2018). Using machine learning to predict student difficulties from learning session data. *Artificial Intelligence Review*, 1-27. doi:https://doi.org/10.1007/s1046

IBM. (2013). IBM SPSS Statistics for Windows, Version 22.0. Armonk, NY: IBM Corp.

Idemudia, E. C., Dasuki, S. I., & Ogedebe, P. (2016). Factors that influence students' programming skills:a case study from a Nigerian university. *International Journal of Quantitative Research in Education, 3*(4), 277–291.

Izu, C., Mirolo, C., & Weerasinghe, A. (2019). Novice Programmers' Reasoning about Reversing Conditional Statements. *Proceedings of the 49th ACM Technical Symposium on Computer Science Education* (pp. 646-651). Baltimore: ACM. doi:10.1145/3159450.3159499

Jackson, L., Lawson, A., Diack, M., Khosravi, E., & Vincent-Finley, R. (2016). An Analysis of Pair Programming as a Pedagogical Tool for Student Success in Computer Science at a Community College. *In Proceedings of EdMedia 2016--World Conference on Educational Media and Technology* (pp. 699-703). Vancouver, BC, Canada: Association for the Advancement of Computing in Education. Retrieved from https://www.learntechlib.org/primary/p/173020/

Jia, P., & Maloney, T. (2015). Using predictive modelling to identify students at risk of poor university outcomes. *Higher Education, 70*(1), 127-149. doi:10.1007/s10734-014-9829-7

Jokhan, A., Sharma, B., & Singh, S. (2018, May 22). Early warning system as a predictor for student performance in higher education blended courses. *Studies in Higher Education*, 1-12. doi:10.1080/03075079.2018.1466872

Jover, J. M., & Ramírez, J. A. (2018). Academic performance, class attendance and seating location of university students in practical lecture. *Journal of Technology and Science Education, 8*(4), 337-345. Retrieved from https://doi.org/10.3926/jotse.353

Kappelman, L., C.Jones, M., Johnson, V., R.Mclean, E., & Bonnme, K. (2016). Skills for success at different stages of an IT professional's career. *Communications of the ACM, 59*(8), pp. 64-70. doi:10.1145/2888391

Kassarnig, V., Bjerre-Nielsen, A., Mones, E., Lehmann, S., & Lassen, D. D. (2017). Class attendance, peer similarity, and academic performance in a large field study. *PloS one, 12*(11). Retrieved from https://doi.org/10.1371/journal.pone.0187078

Kassarnig, V., Mones, E., Bjerre-Nielsen, A., Sapiezynsk, P., Lassen, D. D., & Lehmann, S. (2018). Academic performance and behavioral. *EPJ Data Science, 7*(10), 1-16. Retrieved from https://doi.org/10.1140/epjds/s13688-018-0138-8

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), 566-567. doi:10.1016/j.eururo.2010.11.039

Kori, K., Pedaste, M., Leijen, Ä., & Tõnisson, E. (2016). The role of programming experience in ICT students' learning motivation and academic achievement. *International Journal of Information and Education Technology, 6*(5). doi:10.7763/IJIET.2016.V6.709

Koulouri, T., Lauria, S., & Macredie, R. D. (2015). Teaching Introductory Programming: A Quantitative Evaluation of Different Approaches. (C. Hundhausen, Ed.) *ACM Transactions on Computing Education, 14*(4), 26.1-26.27. doi:10.1145/2662412

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, J. A. Larusson, & B. White (Eds.), *Learning Analytics: From Research to Practice* (pp. 103-119). New York: Springer. doi:10.1007/978-1-4614-3305-7

Kuzilek, J., Hlosta, M., Herrmannova, D., & Zdrahal, Z. a. (2015, March 10). OU Analyse: analysing at-risk students at The Open University. *The Open University. Learning Analytics Review*, pp. 1-16. Retrieved from http://www.laceproject.eu/learning-analyticsreview/analysing-at-risk-students-at-open-university/

Leppänen, L., Leinonen, J., Ihantola, P., & Hellas, A. (2017). Predicting Academic Success Based on Learning Material Usage. *Proceedings of the 18th Annual Conference on Information Technology Education* (pp. 13-18). Rochester, New York: ACM. doi:10.1145/3125659.3125695

Li, J., Cheng, K., Wang, S., Morstatter, F., Trevino, R. p., & Jiliang Tang, H. l. (2017). Feature Selection: A Data Perspective. *ACM Computing Surveys, 50*(6). Retrieved from https://doi.org/10.1145/3136625

Liao, S. N., Zingaro, D., Alvarado, C., Griswold, W. G., & Porter, L. (2019). Exploring the Value of Different Data Sources for Predicting Student Performance in Multiple CS Courses. *Proceedings of the 50th ACM Technical Symposium on Computer Science Education* (pp. 112-118). Minneapolis, MN, USA: ACM. doi:10.1145/3287324.3287407

Liao, S. N., Zingaro, D., Laurenzano, M. A., Griswold, W. G., & Porter, L. (2016). Lightweight, Early Identification of At-Risk CS1 Students. *Proceedings of the 2016 ACM Conference on International Computing Education Research* (pp. 123-131). Melbourne, VIC, Australia: ACM. doi:10.1145/2960310.2960315

Liao, S. N., Zingaro, D., Thai, K., Alvarado, C., Griswold, W. G., & Porter, L. (2019, June). A Robust Machine Learning Technique to Predict Low-performing Students. *ACM Transactions on Computing Education (TOCE), 19*(3), 18:1 - 18:19. doi:10.1145/3277569

Lishinski, A., Yadav, A., & Enbody, R. (2017). Students’ Emotional Reactions to Programming Projects in Introduction to Programming: Measurement Approach and Influence on Learning Outcomes. *Proceedings of the 2017 ACM Conference on International Computing Education Research* (pp. 30-38). Tacoma, WA, USA: ACM. doi:10.1145/3105726.3106187

Lishinski, A., Yadav, A., Enbody, R., & Good, J. (2016). The Influence of Problem Solving Abilities on Students' Performance on Different Assessment Tasks in Introductory Programming. *SIGCSE ’16* (pp. 329-324). Memphis, TN, USA: ACM. doi:10.1145/2839509.2844596

Lister, R., Adams, E. S., Fitzgerald, S., Fone, W., Hamer, J., Morten Lindholm, R. M., . . . Thomas, L. (2004). A Multi-National Study of Reading and Tracing Skills in Novice Programmers. *SIGCSE Bulletin, 36*(4), 119-150.

Longi, K. (2016). *Exploring factors that affect performance on introductory programming courses.* Master’s Thesis, University of Helsinki, Department of Computer Science, Helsinki.

Lukkarinen, A., koivukangas, P., & Seppälä, T. (2016). Relationship between class attendance and student performance. *2nd International Conference on Higher Education Advances. 228*, pp. 341-347. Valencia: Elsevier. doi:10.1016/j.sbspro.2016.07.051

Luxton-Reilly, A., Simon, Albluwi, I., Becker, B. A., Giannakos, M., Kumar, A. N., . . . Szabo, C. (2018). Introductory programming: a systematic literature review. *Proceedings Companion of the 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education* (pp. 55-106). Larnaca: ACM. doi:10.1145/3293881.3295779

Macfadyen, L. P., & Dawson, S. (2010). Mining LMS data to develop an “early warning system” for educators: A proof of concept. *Computers & Education*, 588-599. Retrieved from https://doi.org/10.1016/j.compedu.2009.09.008

Mann, P. S. (2009). Nonparametric Methods. In P. S. Mann, *Introductory Statistics* (pp. 625-650). Wiley.

Mann, P. S. (2009). Simple Linear Regression. In P. S. MANN, *Introductory Statistics* (pp. 564-610). Wiley.

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, 103,* 1-15. Retrieved from http://dx.doi.org/10.1016/j.compedu.2016.09.005

Marling, C., & Juedes, D. (2016). CS0 for Computer Science Majors at Ohio University. *SIGCSE '16 Proceedings of the 47th ACM Technical Symposium on Computing Science Education* (pp. 138-143). New York, NY, USA: ACM. doi:10.1145/2839509.2844624

Miao, J., & Niu, L. (2016). A Survey on Feature Selection. *Procedia Computer Science, 91*, 919-926. Retrieved from https://doi.org/10.1016/j.procs.2016.07.111

Mohamad, S. K., & Tasir, Z. (2013, November 6). Educational Data Mining: A Review. *97*, pp. 320-324. Hilton Kuching, Sarawak: Elsevier. doi:10.1016/j.sbspro.2013.10.240

Morgan, P. (2018). A Machine Learning Report Changing the Rules. In P. Morgan, *Machine Learning Is* (pp. 2-3). Sebastopol: O'Reilly Media, Inc.,.

Moskal, A. C., Gasson, J., & Parsons, D. (2017). The 'Art' of Programming: Exploring Student Conceptions of Programming through the Use of Drawing Methodology. *Proceedings of the 2017 ACM Conference on International Computing Education Research* (pp. 39-46). Tacoma, Washington, USA: ACM. doi:10.1145/3105726.3106170

Mueen, A., Zafar, B., & Manzoor, U. (2016). Modeling and Predicting Students' Academic Performance Using Data Mining Techniques. *I.J. Modern Education and Computer Science, 11*, 36-42. doi:10.5815/ijmecs.2016.11.05

Narula, M., & Nagar, P. (2013). Relationship Between Students' Performance and Class Attendance in a Programming Language Subject in a Computer Course. (M. Zainuddin, Ed.) *International Journal of Computer Science and Mobile Computing, 2*(8), 206-210.

Nivala, M., ParankoHans, J., Gruber, & Lehtinen, E. (2016). The Role of Prior Knowledge and Students' Perceptions in Learning of Biomedical Sciences. *Medical Science Educator, 26*(4), 631-638. doi:10.1007/s40670-016-0319-7

Novianti, P. W., Jong, V. L., Roes, K. C., & Eijkemans, M. J. (2015, June 21). Factors affecting the accuracy of a class prediction model in gene expression data. *BMC Bioinformatics, 16*(199), 1-12. doi:10.1186/s12859-015-0610-4

Osmanbegovic, E., & Suljic, M. (2012). Data mining approach for predicting student performance. *Journal of Economics and Business, X*(1), 3-12.

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

Perez, B., Castellanos, C., & Correal, D. (2018). Applying Data Mining Techniques to Predict Student Dropout: A Case Study. *2018 IEEE 1st Colombian Conference on Applications in Computational Intelligence*. Medellin, Colombia: IEEE. doi:10.1109/ColCACI.2018.8484847

Pistilli, M. D., III, J. E., & Campbell, J. P. (2014). Analytics Through an Institutional lens: Definition, Theory, Design, and Impact. In J. A. Larusson, & B. White, *Learning Analytics From Research to Practice* (pp. 98-99). New York: Springer.

Planchard, M., Daniel, K. L., Maroo, J., Mishra, C., & McLean, T. (2015). Homework, Motivation, and Academic Achievement in a College Genetics Course. *Journal of College Biology Teaching, 41*(2), 11-18.

Psycharis, S., & Kallia, M. (2017). The effects of computer programming on high school students' reasoning skills and mathematical self-efficacy and problem solving. *Instructional Science, 45*, 583–602. doi:10.1007/s11251-017-9421-5

Qian, Y., & Lehman, J. (2017, December). Students' Misconceptions and Other Difficulties in Introductory Programming: A Literature Review. *ACM Transactions on Computing Education, 18*(1), 1-20. doi:10.1145/3077618

Quille, K., & Bergin, S. (2018). Programming: predicting student success early in CS1. a re-validation and replication study. *Proceedings of the 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education* (pp. 15-20). Larnaca, Cyprus: ACM. doi:10.1145/3197091.3197101

Rajoo, M., & Veloo, A. (2015). The Relationship Between Mathematics Homework Engagement And Mathematics Achievement. *Australian Journal of Basic and Applied Sciences, 9*(28), 136-144.

Rix, J., Dewhurst, P., Cooke, C., & Newell, D. (2018). Engagement as predictors of performance in a single cohort of undergraduate chiropractic students. *Journal of Chiropractic Education, 32*(1), 36-42. doi:10.7899/JCE-17-8

Roberts, L. D., Howell, J. A., Seaman, K., & Gibson, D. C. (2016, December). Student Attitudes toward Learning Analytics in Higher Education: "The Fitbit Version of the Learning World". (D. Kauffman, Ed.) *frontiers in Psychology, 7*. doi:10.3389/fpsyg.2016.01959

S., B. B., & Hastings, J. T. (1971). *Handbook on Formative and Summative Evaluation of Student Learning.* New York: McGraw-Hill Book Company.

Shahiri, A. M., Husain, W., & Rashid, N. A. (2015). A Review on Predicting Student's Performance using Data Mining Techniques. *72*, pp. 414-422. Elsevier. doi: 10.1016/j.procs.2015.12.157

Sharma, R., & Shen, D. H. (2018). The Interplay of Factors Affecting Learning of Introductory Programming: A comparative study of an Australian and an Indian University. *The 13th International Conference on Computer Science & Education (ICCSE 2018)* (pp. 669-674). Colombo: IEE.

Silva, M. T. (2014). Failure rates in introductory programming: A 2006–2012 study at a Brazilian University. *IEEE Frontiers in Education Conference (FIE) 2014* (pp. 1-7). IEEE.

Simons, J. M. (2011). *A national study of student early alert models at four-year institutions of higher education.* ProQuest.

Soni, A., & Vivek Kumar, R. K. (2018). Predicting student performance using data mining techniques. *International Journal of Pure and Applied Mathematics*, 221-227. Retrieved from http://www.ijpam.eu

Stewart, B. (2002). Predicting project delivery rates using the Naive–Bayes classifier. *Journal of Software Maintenance and Evolution, 14*, 161-179. doi:10.1002/smr.250

Suarta, I. M., Suwintana, I. K., Sudhana, I. G., & Hariyanti, N. K. (2017). Employability Skills Required by the 21st Century Workplace: A Literature Review of Labor Market Demand. *1st International Conference on Technology and Vocational Teachers (ICTVT 2017). 102*, pp. 337-342. Atlantis Press. Retrieved from http://creativecommons.org/licenses/by-nc/4.0/

Tafliovich, A., Campbell, J., & Petersen, A. (2013). A Student Perspective on Prior Experience in CS1. *Proceeding of the 44th ACM technical symposium on Computer science education* (pp. 239-244). Denver, Colorado, USA.: ACM.

Team, R. C. (2013). *R: A language and environment for statistical computing*. (R Foundation for Statistical Computing) Retrieved from R Foundation for Statistical Computing: http://www.R-project.org/

Thai-Nghe, N., Busche, A., & Schmidt-Thieme, L. (2009). Improving Academic Performance Prediction by Dealing with Class Imbalance. *2009 Ninth*

International Conference on Intelligent Systems Design and Applications (pp. 878-883). IEEE. doi:10.1109/ISDA.2009.15

Timmers, C. F., AmberWalraven, & P.Veldkamp, B. (2015). The effect of regulation feedback in a computer-based formative assessment on information problem solving. *Computers & Education*, *87*(9), 1-9. Retrieved from https://doi.org/10.1016/j.compedu.2015.03.012

Tzu-ChiYang, Chen, M. C., & Y.Chen, S. (2018, November). The influences of self-regulated learning support and prior knowledge on improving learning performance. *Computers & Education, 126*, 37-52. doi:10.1016/j.compedu.2018.06.025

Uysal, M. P. (2014, July 1). 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), 198-217. doi:https://doi.org/10.30935/cedtech/6125

VanDeGrift, T. (2015). Supporting Creativity and User Interaction in CS 1 Homework Assignments. *46th ACM TECHNICAL SYMPOSIUM on COMPUTER SCIENCE EDUCATION* (pp. 54-59). Kansas City: ACM. Retrieved from http://dx.doi.org/10.1145/2676723.2677250

Vandekerckhove, J., Matzke, D., & Wagenmakers, E.-J. (2014). Model Comparison and the Principle of Parsimony. In J. R. Busemeyer, Z. Wang, J. T. Townsend, & A. Eidels (Eds.), *The Oxford Handbook of Computational and Mathematical Psychology* (pp. 300-329). Oxford University Press.

Watson, C., & Li, F. W. (2014). Failure Rates in Introductory Programming Revisited. *Proceedings of the 2014 conference on Innovation & technology in computer science education* (pp. 39-44). Uppsala, Sweden: Association for Computing Machinery. doi:10.1145/2591708.2591749

Watson, C., Li, F. W., & Godwin, J. L. (2014). No tests required: comparing traditional and dynamic predictors of programming success. *Proceedings of the 45th ACM technical symposium on Computer science education* (pp. 469-474). Association for Computing Machinery. doi:10.1145/2538862.2538930

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, 56*(2), 226-253. doi:https://doi.org/10.1177/0735633117707695

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, 35*(2), 246-255. doi:https://doi.org/10.1111/jcal.12326

Veerasamy, A. K., D'Souza, D., Lindén, R., & Laakso, M.-J. (2019, February 1). Prediction of Student Final Exam Performance in an Introductory Programming Course: Development and Validation of the Use of a Support Vector Machine-Regression Model. *Asian Journal of Education and E-learning, 7*(1), 1-14.

Veerasamy, A. K., D'Souza, D., Lindén, R., Kaila, E., Laakso, M.-J., & Salakoski, T. (2016). The Impact of Lecture Attendance on Exams for Novice Programming Students. *International Journal of Modern Education and Computer Science (IJMECS), 8*(5), 1-11. doi:10.5815/ijmecs.2016.05.01

West, D., Luzeckyj, A., Searle, B., Toohey, D., & Price, R. (2018). *The use of learning analytics to support improvements in teaching practice.* Melbourne: Innovative Research Universities.

Yassein, N. A., Helali, R. G., & Mohomad, S. B. (2017). Predicting Student Academic Performance in KSA using Data Mining Techniques. *Journal of Information Technology & Software Engineering, 7*(5), 1-5. doi:10.4172/2165-7866.1000213

Yukselturk, E., & Altiok, S. (2017, May 01). An investigation of the effects of programming with Scratch on the preservice IT teachers' self-efficacy perceptions and attitudes towards computer programming. *British Journal of Educational Technology, 48*(3), 789-801. doi:10.1111/bjet.12453

Yukselturk, E., Ozekes, S., & Türel, Y. K. (2014). Predicting dropout student: An application of data mining methods in an online education program. *European Journal of Open, Distance and e-Learning, 17*(1). doi:10.2478/eurodl-2014-0008

Yurdugül, H., & Aşkar, P. (2013). Learning Programming, Problem Solving and Gender: A Longitudinal Study. *Procedia - Social and Behavioral Sciences. 83,* pp. 605-610. ELSEVIER. Retrieved from https://doi.org/10.1016/j.sbspro.2013.06.115

Zingaro, D., Taylor, C., Porter, L., Clancy, M., Lee, C., Liao, S. N., & Webb, K. C. (2018). Identifying Student Difficulties with Basic Data Structures. *Proceedings of the 2018 ACM Conference on International Computing Education Research* (pp. 169-177). Espoo: ACM. doi:10.1145/3230977.3231005

Özen, Y. (2016). Can I Solve the Problem? A Program Trail on Problem Solving Skill. *American Journal of Applied Psychology, 4*(1), 1-10. doi:DOI:10.12691/ajap-4-1-1

9. Appendices

9.1 Problem Solving Inventory Questionnaire: Finnish version

1. ”Kun ongelman ratkaisu ei onnistunut, en tutki miksi ratkaisu ei toiminut."
2. Kun kohtaan monimutkaisen ongelman, en välitä kehittää strategiaa tiedon keräämiseen, jotta voisin määritellä tarkalleen mikä ongelma on.
3. Kun minun ensimmäiset ongelmanratkaisuyritykseni epäonnistuvat, minulle tulee epämukava olo ajatellessani kykyjäni käsitellä tilannetta.
4. Kun olen ratkaissut ongelman, en analysoi mikä meni oikein ja mikä väärin.
5. Pystyn yleensä keksimään luovia ja tehokkaita vaihtoehtoja ongelman ratkaisemiseksi.
6. Kun olen yrittänyt ratkaista ongelman tietyllä tavalla, käytän aikaa vertaillakseni saavutettuja lopputuloksia alkuperäisiin odotuksiini.
7. Kun minulla on ongelma, yritän keksiä uusia ja uusia tapoja sen ratkaisemiseksi, kunnes en enää keksi enempää.
8. Kohdatessani ongelman, tarkastelen johdonmukaisesti tunteitani selvittääkseni mitä ongelmatilanteessa tapahtuu.
9. Mikä on lempivärisi?
10. Minulla on kyky ratkaista useimmat ongelmat, vaikka mikään ratkaisu ei aluksi olisikaan ilmeinen.
11. Monet ongelmista joita kohtaan ovat liian monimutkaisia ratkaistavakseni.
12. Teen päätöksiä ja olen niihin myöhemmin tyytyväinen.
13. Kohdatessani ongelman, minulla on tapana yrittää ratkaista se ensimmäisellä mieleen tulevalla tavalla.
14. En toisinaan pysähdy ja ota aikaa ratkoakseni ongelmiani, vaan ikäänkuin tarvon eteenpäin.
15. Tehdessäni päätöksiä ideoista tai valitessani ongelman mahdollisista ratkaisuista, en jää pohtimaan miten hyvät mahdollisuudet kullakin vaihtoehdolla on onnistua.
16. Kun kohtaan ongelman, pysähdyn miettimään ennen seuraavaa askelta.
17. Yleensä valitsen ensimmäisen hyvän idean joka mieleeni tulee.
18. Tehdessäni päätöstä, punnitsen jokaisen vaihtoehdon seuraukset ja vertailen niitä keskenään.
19. Kun suunnittelen ratkaisua ongelmaan, olen melkein varma että saan suunnitelman toimimaan.
20. Pyrin ennakoimaan tekemieni toimenpiteiden vaikutuksia tuloksiin.
21. Pyrkiessäni keksimään mahdollisia ratkaisuita ongelmaan, en keksi kovin monia vaihtoehtoja.
22. Millainen päivä sinulla tänään on?
23. Kun aikaa on riittävästi ja yritän tarpeeksi, uskon voivani ratkaista useimmat vastaan tulevat ongelmat."
24. Kun ajaudun uuteen tilanteeseen, olen luottavainen että selviän mahdollisesti kohtaamistani ongelmista.
25. Vaikka teen töitä ongelman ratkaisemiseksi, minusta välillä tuntuu että haparoin tai harhailen, enkä ryhdy ratkaisemaan varsinaista asiaa.
26. Teen päätöksiä hetken mielijohteesta ja kadun niitä myöhemmin.
27. Uskon kykyyni ratkaista uusia ja vaikeita ongelmia.

28. Minulla on systemaattinen tapa vaihtoehtojen vertailuun ja päätösten tekoon.
29. Mikä on lempiruokasi?
30. Kun kohtaan ongelman, en yleensä tutki millaiset ulkoiset asiat ympäristössäni voivat olla ongelman osatekijöitä.
31. Kun en tiedä mitä ongelman kanssa pitäisi tehdä, pyrin ensimmäisten asioiden joukossa kartoittamaan tilanteen ja päättelemään ongelmanratkaisun kannalta olennaiset asiat.
32. Tunteeni ovat välillä niin pinnassa, etten kykene harkitsemaan erilaisia tapoja ongelman ratkaisemiseksi.
33. Tehtyäni päätöksen, odottamani lopputulos yleensä vastaa toteutunutta lopputulosta.
34. Kun kohtaan ongelman, olen epävarma siitä että selviän tilanteesta.
35. Kun tulen tietoiseksi ongelman olemassaolosta, pyrin ensimmäisten asioiden joukossa ratkaisemaan millainen ongelma tarkalleen on.

9.2 Problem Solving Inventory Questionnaire: English version

1. When a solution to a problem was unsuccessful, I do not examine why it didn't work.
2. When I am confronted with a complex problem, I do not bother to develop a strategy to collect information so I can define exactly what the problem is.
3. When my first efforts to solve a problem fail, I become uneasy about my ability to handle the situation.
4. After I have solved a problem, I do not analyse what went right or what went wrong.
5. I am usually able to think up creative and effective alternatives to solve a problem.
6. After I have tried to solve a problem with a certain course of action, I take time and compare the actual outcome to what I thought should have happened.
7. When I have a problem, I think up as many possible ways to handle it as I can until I can't come up with any more ideas.
8. When confronted with a problem, I consistently examine my feelings to find out what is going on in a problem situation.
9. Filler questions
10. I have the ability to solve most problems even though initially no solution is immediately apparent.
11. Many problems I face are too complex for me to solve.
12. I make decisions and am happy with them later.
13. When confronted with a problem, I tend to do the first thing that I can think of to solve it.
14. Sometimes I do not stop and take time to deal with my problems, but just kind of muddle ahead.
15. When deciding on an idea or possible solution to a problem, I do not take time to consider the chances of each alternative being successful.
16. When confronted with a problem, I stop and think about it before deciding on a next step.
17. I generally go with the first good idea that comes to my mind.
18. When making a decision, I weigh the consequences of each alternative and compare them against each other.
19. When I make plans to solve a problem, I am almost certain that I can make them work.
20. I try to predict the overall result of carrying out a particular course of action.

21. When I try to think up possible solutions to a problem, I do not come up with very many alternatives.
22. Filler questions.
23. Given enough time and effort, I believe I can solve most problems that confront.
24. When faced with a novel situation I have confidence that I can handle problem that may arise.
25. Even though I work on a problem, sometimes I feel like I am groping or wandering , and am not getting down to the real issue .
26. I make snap judgments and later regret them.
27. I trust my ability to solve new and difficult problems.
28. I have a systematic method for comparing alternatives and making decision.
29. Filler questions.
30. When confronted with a problem, I do not usually examine what sort of external things my environment may be contributing to my problem.
31. When I am confused by a problem, one of the first things I do is survey the situation and considers all the relevant pieces of information.
32. Sometimes I get so charged up emotionally that I am unable to consider many ways of dealing with my problems.
33. After making a decision, the outcome I expected usually matches the actual outcome.
34. When confronted with a problem, I am unsure of whether I can handle the situation.
35. When I become aware of a problem, one of the first things I do is to try to find out exactly what the problem is.

9.3 Prior Programming knowledge Questionnaire:

<table>
<tr><td colspan="3">Question 1: How much previous programming experience/knowledge (PPK) have you had?</td></tr>
<tr><td>Likert scale</td><td>PPK level</td><td></td></tr>
<tr><td>0</td><td>0</td><td>Means you have no programming experience/ knowledge at all.</td></tr>
<tr><td>1 and 2</td><td>1</td><td>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.</td></tr>
<tr><td>>=3</td><td>2</td><td>Means you have studied one or more programming languages, or you have sufficient knowledge in computer programming. In addition, you know how to write mid-level and or higher-level computer programs.</td></tr>
<tr><td colspan="3">Question 2: Which programming languages have you written over 200 lines of code (note that mark-up languages such as (X) HTML or XML are not counted as programming here)?</td></tr>
</table>

Table PPK. Survey question to examine student's prior programming knowledge- self reported (Author)

9.4 Model prediction accuracies: 15 x 3 terms for *Introduction to Programming*

Table INT_K1. Week 2-K-fold cross-validation results: 2016

No.	MAC	ATSE	ATSP	PPV	NPV	BAC	AUC	At-risk: AUC	At-risk Vs other: CI
#1	46.3	42.86	66.67	45.00	64.71	54.76	0.55	0.55	0.41-0.68
#2	51.85	52.38	63.64	47.83	67.74	58.01	0.58	0.58	0.44-0.72
#3	51.85	66.67	63.64	53.85	75.00	65.15	0.62	0.65	0.52-0.78
#4	31.48	71.43	15.15	34.88	45.45	43.29	0.49	0.44	0.32-0.55
#5	59.26	52.38	84.85	68.75	73.68	68.61	0.60	0.69	0.56-0.81
#6	59.26	57.14	84.85	70.59	75.68	71.00	0.64	0.71	0.59-0.84
#7	44.44	66.67	51.52	46.67	70.83	59.09	0.63	0.59	0.46-0.73
#8	55.56	47.62	81.82	62.50	71.05	64.72	0.61	0.65	0.52-0.78
#9	55.56	52.38	78.79	61.11	72.22	65.58	0.64	0.66	0.53-0.79
#10	50.00	76.19	48.48	48.48	761.9	62.34	0.59	0.62	0.50-0.75
#11	57.41	52.38	78.79	61.11	72.22	65.58	0.59	0.66	0.53-0.66
#12	38.89	71.43	30.30	39.47	62.50	50.87	0.53	0.51	0.38-0.64
#13	55.56	47.62	78.79	58.82	70.27	63.20	0.55	0.63	0.51-0.76
#14	51.85	57.14	57.58	46.15	67.86	57.36	0.58	0.57	0.44-0.71
#15	50.00	52.38	66.67	50.00	68.75	59.52	0.55	0.60	0.46-0.73

Table INT_K2. Week 4-K-fold cross-validation results: 2016

No.	MAC	ATSE	ATSP	PPV	NPV	BAC	AUC	At-risk AUC	At-risk Vs other: CI
#1	46.3	42.86	66.67	45.00	64.71	54.76	0.55	0.55	0.41-0.68
#2	51.85	52.38	63.64	47.83	67.74	58.01	0.58	0.58	0.44-0.72
#3	51.85	66.67	63.64	53.85	75.00	65.15	0.62	0.65	0.52-0.78
#4	42.59	14.29	81.82	33.33	60.00	48.05	0.46	0.48	0.38-0.58
#5	59.25	57.14	78.79	63.16	74.29	67.97	0.58	0.68	0.55-0.81
#6	55.56	53.28	78.79	61.11	72.22	65.58	0.55	0.66	0.53-0.79
#7	42.59	42.86	63.64	42.86	63.64	53.25	0.54	0.53	0.40-0.70
#8	62.96	61.90	81.82	68.42	77.14	71.86	0.58	0.72	0.59-0.84
#9	55.56	52.38	81.82	64.71	72.97	67.10	0.51	0.67	0.54-0.80

No.	MAC	ATSE	ATSP	PPV	NPV	BAC	AUC	At-risk AUC	At-risk Vs other: CI
#10	38.89	42.86	57.58	39.13	61.29	50.22	0.51	0.50	0.36-0.64
#11	64.81	71.43	78.79	68.18	81.25	75.11	0.61	0.75	0.63-0.87
#12	37.04	33.33	60.61	35.50	58.82	46.97	0.65	0.47	0.34-0.60
#13	64.81	71.43	75.76	65.22	80.65	73.59	0.59	0.74	0.61-0.86
#14	55.56	52.38	78.79	61.11	72.22	65.58	0.51	0.66	0.53-0.79
#15	62.96	66.67	81.82	70.00	79.41	74.24	0.58	0.74	0.62-0.87

Table INT_K3. Week 6-K-fold cross-validation results: 2016

No.	MAC	ATSE	ATSP	PPV	NPV	BAC	AUC	At-risk AUC	At-risk Vs other: CI
#1	46.3	42.86	66.67	45.00	64.71	54.76	0.55	0.55	0.41-0.68
#2	51.85	52.38	63.64	47.83	67.74	58.01	0.58	0.58	0.44-0.72
#3	51.85	66.67	63.64	53.85	75.00	65.15	0.62	0.65	0.52-0.78
#4	42.59	19.05	72.73	30.77	58.54	45.89	0.41	0.46	0.34-0.57
#5	57.41	52.38	75.76	57.89	71.43	64.07	0.52	0.64	0.51-0.78
#6	51.85	47.62	69.70	50.00	67.65	58.66	0.61	0.59	0.45-0.72
#7	40.74	33.33	69.70	41.18	62.16	51.52	0.52	0.52	0.39-0.65
#8	55.56	57.14	75.76	60.00	73.53	66.45	0.59	0.67	0.53-0.80
#9	53.70	52.38	72.73	55.00	70.59	62.55	0.53	0.63	0.49-0.76
#10	44.44	47.62	60.61	43.48	64.52	54.11	0.62	0.54	0.40-0.68
#11	57.41	61.90	75.76	61.90	75.76	68.83	0.60	0.69	0.56-0.82
#12	44.44	42.86	57.58	39.13	61.29	50.22	0.61	0.50	0.36-0.64
#13	59.26	57.14	75.76	60.00	73.53	66.45	0.53	0.67	0.53-0.80
#14	55.56	57.14	72.73	57.14	72.73	64.94	0.63	0.65	0.52-0.78
#15	51.85	57.14	72.73	57.14	72.73	64.94	0.54	0.65	0.52-0.78

Table INT_V1. Week 2-Validation results: 2017

No.	MAC	ATSE	ATSP	PPV	NPV	BAC	AUC	At-risk AUC	At-risk Vs other: CI
#1	25.00	12.50	53.85	7.70	66.67	33.17	0.58	0.33	0.22-0.44
#2	41.18	81.25	44.23	30.95	88.46	62.74	0.57	0.63	0.51-0.75
#3	38.24	68.75	46.15	28.21	82.76	57.45	0.56	0.58	0.44-0.71
#4	22.06	75.00	11.54	20.69	60.00	43.27	0.46	0.44	0.31-0.55
#5	41.18	37.50	61.54	23.08	76.19	49.52	0.58	0.50	0.36-0.63
#6	44.12	50.00	59.62	27.59	79.49	54.81	0.59	0.55	0.40-0.69
#7	29.41	68.75	32.69	23.91	77.27	50.72	0.57	0.51	0.37-0.64
#8	42.65	37.50	61.54	23.07	76.19	49.52	0.57	0.50	0.36-0.64
#9	47.06	56.25	57.69	29.03	81.08	56.97	0.58	0.57	0.43-0.72
#10	30.88	56.25	34.62	20.93	72.00	45.43	0.48	0.45	0.31-0.60
#11	41.18	50.00	55.77	25.81	78.38	52.88	0.54	0.53	0.39-0.67
#12	25.00	62.50	21.15	19.61	64.71	41.83	0.44	0.42	0.28-0.53
#13	44.12	50.00	61.54	28.57	80.00	55.77	0.58	0.56	0.41-0.70
#14	32.35	75.00	25.00	23.53	76.47	50.00	0.46	0.5	0.38-0.62
#15	36.76	68.75	36.54	25.00	79.17	52.64	0.52	0.53	0.39-0.66

Table INT_V2. Week 4-Validation results: 2017

No.	MAC	ATSE	ATSP	PPV	NPV	BAC	AUC	At-risk AUC	At-risk Vs other: CI
#1	25.00	12.50	53.85	7.70	66.67	33.17	0.58	0.33	0.22-0.44
#2	41.18	81.25	44.23	30.95	88.46	62.74	0.57	0.63	0.51-0.75
#3	38.24	68.75	46.15	28.21	82.76	57.45	0.56	0.58	0.44-0.71
#4	45.59	18.75	86.54	30.00	77.59	52.64	0.48	0.53	0.42-0.64
#5	45.59	62.50	50.00	27.78	81.25	56.25	0.60	0.56	0.42-0.70
#6	47.06	62.50	53.85	29.41	82.35	58.17	0.60	0.58	0.44-0.72
#7	30.88	18.75	61.54	13.04	71.11	40.14	0.46	0.40	0.28-0.52
#8	48.53	62.50	55.77	30.30	82.86	59.13	0.60	0.59	0.45-0.73
#9	48.53	62.50	59.62	32.26	83.78	61.06	0.60	0.61	0.47-0.75
#10	35.29	62.50	48.08	27.03	80.65	55.29	0.55	0.55	0.41-0.69

No.	MAC	ATSE	ATSP	PPV	NPV	BAC	AUC	At-risk AUC	At-risk Vs other: CI
#11	54.41	81.25	55.77	36.11	90.62	68.51	0.65	0.69	0.57-0.81
#12	36.76	75.00	40.38	27.91	84.00	57.69	0.52	0.58	0.45-0.71
#13	50.00	75.00	50.00	31.58	86.67	23.53	0.66	0.63	0.50-0.75
#14	50.00	68.75	55.77	32.35	85.29	62.26	0.63	0.62	0.49-0.76
#15	54.41	75.00	59.62	36.36	88.57	67.31	0.63	0.67	0.55-0.80

Table INT_V3. Week 6-Validation results: 2017

No.	MAC	ATSE	ATSP	PPV	NPV	BAC	AUC	At-risk AUC	At-risk Vs other: CI
#1	25.00	12.50	53.85	7.70	66.67	33.17	0.58	0.33	0.22-0.44
#2	41.18	81.25	44.23	30.95	88.46	62.74	0.57	0.63	0.51-0.75
#3	38.24	68.75	46.15	28.21	82.76	57.45	0.56	0.58	0.44-0.71
#4	51.47	37.50	84.62	42.86	81.48	61.06	0.49	0.61	0.48-0.74
#5	48.53	81.25	50.00	33.33	89.66	65.62	0.65	0.66	0.54-0.78
#6	50.00	68.75	61.54	35.48	86.49	65.14	0.63	0.65	0.52-0.79
#7	50.00	43.75	78.85	38.89	82.00	61.30	0.58	0.61	0.48-0.75
#8	51.47	68.75	59.62	34.38	86.11	64.18	0.58	0.64	0.51-0.78
#9	51.47	68.75	63.46	36.67	86.84	66.11	0.58	0.66	0.53-0.80
#10	39.71	62.50	53.85	29.41	82.35	58.17	0.58	0.58	0.44-0.72
#11	51.47	82.25	50.00	33.33	89.66	65.62	0.63	0.66	0.54-0.78
#12	36.76	75.00	40.38	27.91	84.00	57.69	0.52	0.58	0.45-0.71
#13	47.06	81.25	46.15	31.71	88.89	63.70	0.64	0.64	0.52-0.76
#14	48.53	68.75	57.69	33.33	85.71	63.22	0.62	0.63	0.50-0.77
#15	51.47	68.75	59.62	34.38	86.11	64.18	0.59	0.64	0.51-0.78

Table INT_T1. Week 2- Unknown data test results: 2018

No.	MAC	ATSE	ATSP	PPV	NPV	BAC	AUC	At-risk AUC	At-risk Vs other: CI
#1	34.92	37.93	44.12	36.67	45.45	41.02	0.60	0.59	0.47-0.71
#2	44.44	44.83	61.76	50.00	56.76	53.30	0.56	0.53	0.41-0.66
#3	38.1	44.83	44.12	40.62	48.39	44.47	0.58	0.56	0.43-0.68
#4	44.44	96.55	00.00	45.16	00.00	48.28	0.49	0.48	0.45-0.52
#5	52.38	82.76	38.24	53.33	72.22	60.50	0.55	0.61	0.50-0.71
#6	50.79	82.76	35.29	52.17	70.59	59.03	0.55	0.59	0.48-0.70
#7	36.51	65.52	17.65	40.43	37.50	41.58	0.44	0.42	0.30-0.53
#8	47.62	72.41	41.18	51.22	63.64	56.80	0.54	0.57	0.45-0.69
#9	50.79	75.86	38.24	51.16	65.00	57.05	0.53	0.57	0.46-0.69
#10	31.75	51.72	23.53	36.59	36.36	37.63	0.42	0.38	0.26-0.49
#11	46.03	72.41	32.35	47.73	57.89	52.38	0.48	0.52	0.41-0.64
#12	41.27	82.76	8.82	43.64	37.50	45.79	0.47	0.46	0.37-0.54
#13	55.56	82.76	47.06	57.14	76.19	64.91	0.61	0.65	0.54-0.76
#14	50.79	86.21	23.53	49.02	66.67	54.87	0.48	0.55	0.45-0.65
#15	50.79	79.31	29.41	48.94	62.50	54.36	0.49	0.54	0.44-0.65

Table INT_T2. Week 4- Unknown data test results: 2018

No.	MAC	ATSE	ATSP	PPV	NPV	BAC	AUC	At-risk AUC	At-risk Vs other: CI
#1	34.92	37.93	44.12	36.67	45.45	41.02	0.60	0.59	0.47-0.71
#2	44.44	44.83	61.76	50.00	56.76	53.30	0.56	0.53	0.41-0.66
#3	38.1	44.83	44.12	40.62	48.39	44.47	0.58	0.56	0.43-0.68
#4	38.10	13.79	94.11	66.67	56.14	53.96	0.55	0.54	0.46-0.64
#5	49.21	86.21	26.47	50.00	69.23	56.34	0.54	0.56	0.47-0.66
#6	49.21	86.21	26.47	50.00	69.23	56.34	0.54	0.56	0.47-0.66
#7	30.16	27.59	44.12	29.63	41.67	35.85	0.63	0.64	0.52-0.76
#8	47.62	75.86	35.29	50.00	63.16	55.58	0.52	0.56	0.44-0.70
#9	47.62	72.41	38.24	50.00	61.90	55.32	0.51	0.55	0.44-0.67

No.	MAC	ATSE	ATSP	PPV	NPV	BAC	AUC	At-risk AUC	At-risk Vs other: CI
#10	36.51	41.38	44.12	38.71	46.88	42.75	0.58	0.57	0.45-0.70
#11	47.62	72.41	35.29	48.84	60.00	53.85	0.48	0.54	0.42-0.66
#12	47.62	51.72	61.76	61.76	53.57	60.00	56.74	0.57	0.44-0.69
#13	49.21	79.31	32.35	50.00	64.71	55.83	0.51	0.56	0.45-0.67
#14	47.62	72.41	38.24	50.00	61.90	55.32	0.51	0.55	0.44-0.67
#15	46.03	68.97	38.24	48.78	59.09	53.60	0.50	0.54	0.42-0.66

Table INT_T3. Week 6- Unknown data test results: 2018

No.	MAC	ATSE	ATSP	PPV	NPV	BAC	AUC	At-risk AUC	At-risk Vs other: CI
#1	34.92	37.93	44.12	36.67	45.45	41.02	0.60	0.59	0.47-0.71
#2	44.44	44.83	61.76	50.00	56.76	53.30	0.56	0.53	0.41-0.66
#3	38.1	44.83	44.12	40.62	48.39	44.47	0.58	0.56	0.43-0.68
#4	42.86	31.03	82.35	60.00	58.33	56.69	51.94	0.57	0.46-0.68
#5	61.90	75.86	67.65	66.67	76.67	71.75	0.68	0.72	0.61-0.83
#6	53.97	58.62	70.59	62.96	66.67	64.60	0.52	0.65	0.53-0.77
#7	34.92	27.59	61.76	38.10	50.00	44.68	0.43	0.45	0.33-0.56
#8	53.97	65.52	58.82	57.58	66.67	62.17	0.59	0.62	0.50-0.74
#9	50.79	51.72	70.59	60.00	63.16	61.16	0.49	0.61	0.49-0.73
#10	41.27	44.83	52.94	44.83	52.94	48.88	0.56	0.49	0.36-0.61
#11	50.79	65.52	52.94	54.29	64.29	59.23	0.57	0.59	0.47-0.72
#12	47.62	58.62	52.94	51.52	60.00	55.78	0.55	0.56	0.43-0.68
#13	53.97	75.86	52.94	57.89	72.00	64.40	0.60	0.64	0.53-0.76
#14	58.73	65.52	73.53	67.86	67.86	71.43	0.55	0.70	0.58-0.81
#15	50.79	58.62	61.76	56.67	63.64	60.19	0.55	0.60	0.48-0.73

9.5 Model prediction accuracies: 15 X 2 terms for *Algorithms and Programming*

Table ALG_K1. Week 2-K-fold cross-validation results: 2016

No.	MAC	ATSE	ATSP	PPV	NPV	BAC	AUC	At-risk AUC	At-risk Vs other: CI
#16	44.12	22.73	89.68	43.48	76.87	56.21	0.51	0.56	0.49-0.63
#17	44.12	4.55	93.65	20.00	73.75	49.10	0.44	0.49	0.45-0.53
#18	44.71	11.36	90.48	29.41	74.51	50.92	0.47	0.51	0.46-0.56
#19	38.82	9.09	88.89	22.22	73.68	48.99	0.50	0.49	0.44-0.54
#20	30.59	93.18	13.49	27.33	85.00	53.34	0.54	0.53	0.49-0.58
#21	36.47	22.73	73.01	22.73	73.02	47.87	0.51	0.48	0.41-0.55
#22	38.82	9.91	88.89	22.22	73.68	48.99	0.50	0.49	0.44-0.54
#23	34.12	90.91	20.63	28.57	86.67	55.77	0.54	0.56	0.50-0.61
#24	42.35	43.18	69.84	33.33	77.88	56.51	0.54	0.57	0.48-0.65
#25	42.94	36.36	78.57	37.21	77.95	57.47	0.54	0.57	0.49-0.66
#26	41.76	84.09	35.71	31.36	86.54	59.90	0.55	0.60	0.53-0.67
#27	42.35	36.36	76.19	34.78	77.42	56.28	0.53	0.56	0.48-0.64
#28	34.12	90.91	21.43	28.78	87.10	56.17	0.55	0.56	0.51-0.62
#29	45.29	40.91	80.16	41.86	79.53	60.53	0.58	0.61	0.52-0.69
#30	45.88	47.73	76.13	41.18	80.67	61.96	0.58	0.62	0.54-0.70

Table ALG_K2. Week 3-K-fold cross-validation results: 2016

No.	MAC	ATSE	ATSP	PPV	NPV	BAC	AUC	At-risk AUC	At-risk Vs other: CI
#19	44.71	00.00	96.03	00.00	73.33	48.02	0.49	0.48	0.46-0.50
#20	42.94	00.00	100.0	0	74.12	50.00	0.51	0.50	0.50-0.50
#21	45.29	4.55	94.44	22.22	73.91	49.49	0.51	0.50	0.46-0.53
#22	44.71	11.34	90.48	29.41	74.51	50.92	0.53	0.51	0.46-0.56
#23	44.71	15.91	92.06	41.18	75.82	53.99	0.52	0.54	0.48-0.60
#24	43.53	6.82	91.27	21.43	73.72	49.04	0.50	0.49	0.45-0.54

No.	MAC	ATSE	ATSP	PPV	NPV	BAC	AUC	At-risk AUC	At-risk Vs other: CI
#25	48.24	36.36	80.95	40.00	78.46	58.66	0.55	0.59	0.51-0.67
#26	44.71	25.00	85.71	37.93	76.60	55.36	0.49	0.55	0.48-0.63
#27	48.82	31.82	83.33	40.00	77.78	57.58	0.53	0.58	0.50-0.65
#28	47.65	31.82	85.71	43.75	78.26	58.77	0.53	0.59	0.51-0.66
#29	47.06	34.09	81.75	39.47	78.03	57.92	0.54	0.58	0.50-0.66
#30	47.65	36.36	81.75	41.03	78.63	59.06	0.55	0.59	0.51-0.67

Table ALG_K3. Week 4-K-fold cross-validation results: 2016

No.	MAC	ATSE	ATSP	PPV	NPV	BAC	AUC	At-risk AUC	At-risk Vs other: CI
#16	44.12	22.73	89.68	43.48	76.87	56.21	0.51	0.56	0.49-0.63
#17	44.12	4.55	93.65	20.00	73.75	49.10	0.44	0.49	0.45-0.53
#18	44.71	11.36	90.48	29.41	74.51	50.92	0.47	0.51	0.46-0.56
#19	49.41	9.09	96.83	50.00	75.31	52.96	0.52	0.53	0.48-0.58
#20	41.76	00.00	100.0	NaN	74.12	50.00	0.51	0.50	0.50-0.50
#21	46.47	9.09	95.24	40.00	75.00	52.17	0.51	0.52	0.48-0.57
#22	46.47	6.82	93.65	27.27	74.21	50.23	0.49	0.50	0.46-0.55
#23	44.71	25.00	89.68	45.83	77.40	57.34	0.53	0.57	0.50-0.60
#24	45.29	11.36	93.65	38.46	75.16	52.51	0.52	0.53	0.47-0.58
#25	49.41	31.82	84.13	41.18	77.94	57.97	0.51	0.58	0.50-0.66
#26	44.71	25.00	85.71	37.93	76.60	55.36	0.52	0.55	0.48-0.63
#27	51.18	34.09	83.33	41.67	78.36	58.71	0.53	0.59	0.51-0.67
#28	45.29	29.55	84.13	39.39	77.37	56.84	0.52	0.57	0.49-0.64
#29	48.24	34.09	83.33	41.67	78.36	58.71	0.53	0.59	0.51-0.67
#30	48.82	36.36	84.13	44.44	79.10	60.25	0.56	0.60	0.52-0.68

Table ALG_T3. Week 2- unknown data results: 2017

No.	MAC	ATSE	ATSP	PPV	NPV	BAC	overall AUC	At-risk AUC	At-risk Vs other: CI
#16	53.1	60.71	70.94	33.33	88.3	65.83	0.54	0.66	0.56-0.76
#17	51.72	0	100	0	80.69	50	0.62	0.5	0.50-0.50
#18	46.21	46.43	72.65	28.89	85	59.54	0.52	0.6	0.49-0.70
#19	54.48	7.14	98.29	50	81.56	52.72	0.53	0.53	0.48-0.58
#20	21.38	82.14	8.55	17.69	66.67	45.35	0.46	0.45	0.38-0.53
#21	53.1	0	98.29	0	80.42	49.15	0.49	0.49	0.48-0.50
#22	54.48	7.14	98.29	50	81.56	52.72	0.53	0.53	0.48-0.58
#23	34.48	64.29	35.9	19.35	80.77	50.09	0.46	0.5	0.40-0.60
#24	55.86	28.57	89.74	40	84	59.16	0.54	0.59	0.50-0.68
#25	43.45	60.71	45.3	20.99	82.81	53.01	0.54	0.53	0.43-0.63
#26	37.24	71.43	35.04	20.83	83.67	53.24	0.44	0.53	0.43-0.63
#27	41.38	60.71	41.88	20	81.67	51.3	0.53	0.51	0.41-0.62
#28	37.93	57.14	42.74	19.28	80.65	49.94	0.54	0.5	0.40-0.60
#29	41.38	50	46.15	18.18	79.41	48.08	0.56	0.52	0.42-0.62
#30	42.76	57.14	46.15	20.25	81.82	51.65	0.52	0.52	0.42-0.62

Table ALG_T3. Week 3- unknown data results: 2017

No.	MAC	ATSE	ATSP	PPV	NPV	BAC	overall AUC	At-risk: AUC	At-risk Vs other: CI
#16	53.1	60.71	70.94	33.33	88.3	65.83	0.54	0.66	0.56-0.76
#17	51.72	0	100	0	80.69	50	0.62	0.5	0.50-0.50
#18	46.21	46.43	72.65	28.89	85	59.54	0.52	0.6	0.49-0.70
#19	50.34	0	99.15	0	80.56	49.57	0.52	0.5	0.49-0.50
#20	50.34	0	99.14	80.56	49.57	50	0.52	0.5	0.49-0.50
#21	49.66	0	100.00	0	80.69	50	0.48	0.5	0.50-0.50
#22	55.86	50	84.62	43.75	87.61	67.31	0.58	0.67	0.57-0.77
#23	54.48	39.29	85.47	39.29	85.47	62.38	0.52	0.62	0.53-0.72
#24	48.97	17.86	88.89	27.78	81.89	53.37	0.48	0.53	0.46-0.61
#25	40.69	71.43	41.88	22.73	85.96	56.65	0.57	0.57	0.47-0.66
#26	39.31	50	51.28	19.72	81.08	50.64	0.59	0.6	0.54-0.68
#27	38.62	60.71	41.88	20	81.67	51.3	0.54	0.51	0.41-0.62

No.	MAC	ATSE	ATSP	PPV	NPV	BAC	overall AUC	At-risk:	At-risk Vs other: CI
#28	38.62	60.71	41.88	20	81.67	51.3	0.54	0.51	0.41-0.62
#29	36.55	57.14	41.88	19.05	80.33	49.51	0.52	0.5	0.39-0.60
#30	37.24	60.71	41.88	20	81.67	51.3	0.53	0.51	0.41-0.62

Table ALG_T3. Week 4- unknown data results: 2017

No.	MAC	ATSE	ATSP	PPV	NPV	BAC	AUC	At-risk AUC	At-risk Vs other: CI
#16	53.10	60.71	70.94	33.33	88.30	65.83	0.54	0.66	0.56-0.76
#17	51.72	00.00	100.0	0	80.69	50.00	0.62	0.50	0.50-0.50
#18	46.21	46.43	72.65	28.89	85.00	59.54	0.52	0.60	0.49-0.70
#19	53.10	14.29	95.73	44.44	82.35	55.01	0.52	0.55	0.48-0.62
#20	46.90	00.00	100.0	Nan	80.69	50.00	0.46	0.50	0.50-0.50
#21	50.34	17.86	95.73	50.00	82.96	56.79	0.51	0.57	0.49-0.64
#22	53.10	21.43	90.60	35.29	82.81	56.01	0.51	0.56	0.48-0.64
#23	51.03	50.00	78.63	35.89	86.79	64.32	0.51	0.64	0.54-0.75
#24	49.66	21.43	90.60	35.29	82.81	56.01	0.48	0.56	0.49-0.64
#25	40.00	53.57	51.28	20.83	82.19	52.43	0.54	0.52	0.42-0.63
#26	36.55	53.57	47.01	19.48	80.88	50.29	0.52	0.50	0.40-0.61
#27	39.31	11.36	94.06	45.46	70.90	52.71	0.54	0.52	0.42-0.62
#28	34.48	60.71	39.32	19.32	80.70	50.02	0.53	0.50	0.40-0.60
#29	36.55	60.71	42.74	20.24	81.97	51.72	0.53	0.52	0.42-0.62
#30	37.24	60.71	43.59	20.48	82.26	52.15	0.54	0.52	0.42-0.62

Turku Centre for Computer Science
TUCS Dissertations

1. **Marjo Lipponen**, On Primitive Solutions of the Post Correspondence Problem
2. **Timo Käkölä**, Dual Information Systems in Hyperknowledge Organizations
3. **Ville Leppänen**, Studies on the Realization of PRAM
4. **Cunsheng Ding**, Cryptographic Counter Generators
5. **Sami Viitanen**, Some New Global Optimization Algorithms
6. **Tapio Salakoski**, Representative Classification of Protein Structures
7. **Thomas Långbacka**, An Interactive Environment Supporting the Development of Formally Correct Programs
8. **Thomas Finne**, A Decision Support System for Improving Information Security
9. **Valeria Mihalache**, Cooperation, Communication, Control. Investigations on Grammar Systems.
10. **Marina Waldén**, Formal Reasoning About Distributed Algorithms
11. **Tero Laihonen**, Estimates on the Covering Radius When the Dual Distance is Known
12. **Lucian Ilie**, Decision Problems on Orders of Words
13. **Jukkapekka Hekanaho**, An Evolutionary Approach to Concept Learning
14. **Jouni Järvinen**, Knowledge Representation and Rough Sets
15. **Tomi Pasanen**, In-Place Algorithms for Sorting Problems
16. **Mika Johnsson**, Operational and Tactical Level Optimization in Printed Circuit Board Assembly
17. **Mats Aspnäs**, Multiprocessor Architecture and Programming: The Hathi-2 System
18. **Anna Mikhajlova**, Ensuring Correctness of Object and Component Systems
19. **Vesa Torvinen**, Construction and Evaluation of the Labour Game Method
20. **Jorma Boberg**, Cluster Analysis. A Mathematical Approach with Applications to Protein Structures
21. **Leonid Mikhajlov**, Software Reuse Mechanisms and Techniques: Safety Versus Flexibility
22. **Timo Kaukoranta**, Iterative and Hierarchical Methods for Codebook Generation in Vector Quantization
23. **Gábor Magyar**, On Solution Approaches for Some Industrially Motivated Combinatorial Optimization Problems
24. **Linas Laibinis**, Mechanised Formal Reasoning About Modular Programs
25. **Shuhua Liu**, Improving Executive Support in Strategic Scanning with Software Agent Systems
26. **Jaakko Järvi**, New Techniques in Generic Programming – C++ is more Intentional than Intended
27. **Jan-Christian Lehtinen**, Reproducing Kernel Splines in the Analysis of Medical Data
28. **Martin Büchi**, Safe Language Mechanisms for Modularization and Concurrency
29. **Elena Troubitsyna**, Stepwise Development of Dependable Systems
30. **Janne Näppi**, Computer-Assisted Diagnosis of Breast Calcifications
31. **Jianming Liang**, Dynamic Chest Images Analysis
32. **Tiberiu Seceleanu**, Systematic Design of Synchronous Digital Circuits
33. **Tero Aittokallio**, Characterization and Modelling of the Cardiorespiratory System in Sleep-Disordered Breathing
34. **Ivan Porres**, Modeling and Analyzing Software Behavior in UML
35. **Mauno Rönkkö**, Stepwise Development of Hybrid Systems
36. **Jouni Smed**, Production Planning in Printed Circuit Board Assembly
37. **Vesa Halava**, The Post Correspondence Problem for Market Morphisms
38. **Ion Petre**, Commutation Problems on Sets of Words and Formal Power Series
39. **Vladimir Kvassov**, Information Technology and the Productivity of Managerial Work
40. **Frank Tétard**, Managers, Fragmentation of Working Time, and Information Systems

41. **Jan Manuch**, Defect Theorems and Infinite Words
42. **Kalle Ranto**, Z_4-Goethals Codes, Decoding and Designs
43. **Arto Lepistö**, On Relations Between Local and Global Periodicity
44. **Mika Hirvensalo**, Studies on Boolean Functions Related to Quantum Computing
45. **Pentti Virtanen**, Measuring and Improving Component-Based Software Development
46. **Adekunle Okunoye**, Knowledge Management and Global Diversity – A Framework to Support Organisations in Developing Countries
47. **Antonina Kloptchenko**, Text Mining Based on the Prototype Matching Method
48. **Juha Kivijärvi**, Optimization Methods for Clustering
49. **Rimvydas Rukšėnas**, Formal Development of Concurrent Components
50. **Dirk Nowotka**, Periodicity and Unbordered Factors of Words
51. **Attila Gyenesei**, Discovering Frequent Fuzzy Patterns in Relations of Quantitative Attributes
52. **Petteri Kaitovaara**, Packaging of IT Services – Conceptual and Empirical Studies
53. **Petri Rosendahl**, Niho Type Cross-Correlation Functions and Related Equations
54. **Péter Majlender**, A Normative Approach to Possibility Theory and Soft Decision Support
55. **Seppo Virtanen**, A Framework for Rapid Design and Evaluation of Protocol Processors
56. **Tomas Eklund**, The Self-Organizing Map in Financial Benchmarking
57. **Mikael Collan**, Giga-Investments: Modelling the Valuation of Very Large Industrial Real Investments
58. **Dag Björklund**, A Kernel Language for Unified Code Synthesis
59. **Shengnan Han**, Understanding User Adoption of Mobile Technology: Focusing on Physicians in Finland
60. **Irina Georgescu**, Rational Choice and Revealed Preference: A Fuzzy Approach
61. **Ping Yan**, Limit Cycles for Generalized Liénard-Type and Lotka-Volterra Systems
62. **Joonas Lehtinen**, Coding of Wavelet-Transformed Images
63. **Tommi Meskanen,** On the NTRU Cryptosystem
64. **Saeed Salehi**, Varieties of Tree Languages
65. **Jukka Arvo**, Efficient Algorithms for Hardware-Accelerated Shadow Computation
66. **Mika Hirvikorpi**, On the Tactical Level Production Planning in Flexible Manufacturing Systems
67. **Adrian Costea**, Computational Intelligence Methods for Quantitative Data Mining
68. **Cristina Seceleanu**, A Methodology for Constructing Correct Reactive Systems
69. **Luigia Petre**, Modeling with Action Systems
70. **Lu Yan**, Systematic Design of Ubiquitous Systems
71. **Mehran Gomari**, On the Generalization Ability of Bayesian Neural Networks
72. **Ville Harkke**, Knowledge Freedom for Medical Professionals – An Evaluation Study of a Mobile Information System for Physicians in Finland
73. **Marius Cosmin Codrea**, Pattern Analysis of Chlorophyll Fluorescence Signals
74. **Aiying Rong**, Cogeneration Planning Under the Deregulated Power Market and Emissions Trading Scheme
75. **Chihab BenMoussa**, Supporting the Sales Force through Mobile Information and Communication Technologies: Focusing on the Pharmaceutical Sales Force
76. **Jussi Salmi**, Improving Data Analysis in Proteomics
77. **Orieta Celiku**, Mechanized Reasoning for Dually-Nondeterministic and Probabilistic Programs
78. **Kaj-Mikael Björk**, Supply Chain Efficiency with Some Forest Industry Improvements
79. **Viorel Preoteasa**, Program Variables – The Core of Mechanical Reasoning about Imperative Programs
80. **Jonne Poikonen**, Absolute Value Extraction and Order Statistic Filtering for a Mixed-Mode Array Image Processor
81. **Luka Milovanov**, Agile Software Development in an Academic Environment
82. **Francisco Augusto Alcaraz Garcia**, Real Options, Default Risk and Soft Applications
83. **Kai K. Kimppa**, Problems with the Justification of Intellectual Property Rights in Relation to Software and Other Digitally Distributable Media
84. **Dragoş Truşcan**, Model Driven Development of Programmable Architectures
85. **Eugen Czeizler**, The Inverse Neighborhood Problem and Applications of Welch Sets in Automata Theory

86. **Sanna Ranto**, Identifying and Locating-Dominating Codes in Binary Hamming Spaces
87. **Tuomas Hakkarainen**, On the Computation of the Class Numbers of Real Abelian Fields
88. **Elena Czeizler**, Intricacies of Word Equations
89. **Marcus Alanen**, A Metamodeling Framework for Software Engineering
90. **Filip Ginter**, Towards Information Extraction in the Biomedical Domain: Methods and Resources
91. **Jarkko Paavola**, Signature Ensembles and Receiver Structures for Oversaturated Synchronous DS-CDMA Systems
92. **Arho Virkki**, The Human Respiratory System: Modelling, Analysis and Control
93. **Olli Luoma**, Efficient Methods for Storing and Querying XML Data with Relational Databases
94. **Dubravka Ilić**, Formal Reasoning about Dependability in Model-Driven Development
95. **Kim Solin**, Abstract Algebra of Program Refinement
96. **Tomi Westerlund**, Time Aware Modelling and Analysis of Systems-on-Chip
97. **Kalle Saari**, On the Frequency and Periodicity of Infinite Words
98. **Tomi Kärki**, Similarity Relations on Words: Relational Codes and Periods
99. **Markus M. Mäkelä**, Essays on Software Product Development: A Strategic Management Viewpoint
100. **Roope Vehkalahti**, Class Field Theoretic Methods in the Design of Lattice Signal Constellations
101. **Anne-Maria Ernvall-Hytönen**, On Short Exponential Sums Involving Fourier Coefficients of Holomorphic Cusp Forms
102. **Chang Li**, Parallelism and Complexity in Gene Assembly
103. **Tapio Pahikkala**, New Kernel Functions and Learning Methods for Text and Data Mining
104. **Denis Shestakov**, Search Interfaces on the Web: Querying and Characterizing
105. **Sampo Pyysalo**, A Dependency Parsing Approach to Biomedical Text Mining
106. **Anna Sell**, Mobile Digital Calendars in Knowledge Work
107. **Dorina Marghescu**, Evaluating Multidimensional Visualization Techniques in Data Mining Tasks
108. **Tero Säntti**, A Co-Processor Approach for Efficient Java Execution in Embedded Systems
109. **Kari Salonen**, Setup Optimization in High-Mix Surface Mount PCB Assembly
110. **Pontus Boström**, Formal Design and Verification of Systems Using Domain-Specific Languages
111. **Camilla J. Hollanti**, Order-Theoretic Mehtods for Space-Time Coding: Symmetric and Asymmetric Designs
112. **Heidi Himmanen**, On Transmission System Design for Wireless Broadcasting
113. **Sébastien Lafond**, Simulation of Embedded Systems for Energy Consumption Estimation
114. **Evgeni Tsivtsivadze**, Learning Preferences with Kernel-Based Methods
115. **Petri Salmela**, On Commutation and Conjugacy of Rational Languages and the Fixed Point Method
116. **Siamak Taati**, Conservation Laws in Cellular Automata
117. **Vladimir Rogojin**, Gene Assembly in Stichotrichous Ciliates: Elementary Operations, Parallelism and Computation
118. **Alexey Dudkov**, Chip and Signature Interleaving in DS CDMA Systems
119. **Janne Savela**, Role of Selected Spectral Attributes in the Perception of Synthetic Vowels
120. **Kristian Nybom**, Low-Density Parity-Check Codes for Wireless Datacast Networks
121. **Johanna Tuominen**, Formal Power Analysis of Systems-on-Chip
122. **Teijo Lehtonen**, On Fault Tolerance Methods for Networks-on-Chip
123. **Eeva Suvitie**, On Inner Products Involving Holomorphic Cusp Forms and Maass Forms
124. **Linda Mannila**, Teaching Mathematics and Programming – New Approaches with Empirical Evaluation
125. **Hanna Suominen**, Machine Learning and Clinical Text: Supporting Health Information Flow
126. **Tuomo Saarni**, Segmental Durations of Speech
127. **Johannes Eriksson**, Tool-Supported Invariant-Based Programming

128. **Tero Jokela**, Design and Analysis of Forward Error Control Coding and Signaling for Guaranteeing QoS in Wireless Broadcast Systems
129. **Ville Lukkarila**, On Undecidable Dynamical Properties of Reversible One-Dimensional Cellular Automata
130. **Qaisar Ahmad Malik**, Combining Model-Based Testing and Stepwise Formal Development
131. **Mikko-Jussi Laakso**, Promoting Programming Learning: Engagement, Automatic Assessment with Immediate Feedback in Visualizations
132. **Riikka Vuokko**, A Practice Perspective on Organizational Implementation of Information Technology
133. **Jeanette Heidenberg**, Towards Increased Productivity and Quality in Software Development Using Agile, Lean and Collaborative Approaches
134. **Yong Liu**, Solving the Puzzle of Mobile Learning Adoption
135. **Stina Ojala**, Towards an Integrative Information Society: Studies on Individuality in Speech and Sign
136. **Matteo Brunelli**, Some Advances in Mathematical Models for Preference Relations
137. **Ville Junnila**, On Identifying and Locating-Dominating Codes
138. **Andrzej Mizera**, Methods for Construction and Analysis of Computational Models in Systems Biology. Applications to the Modelling of the Heat Shock Response and the Self-Assembly of Intermediate Filaments.
139. **Csaba Ráduly-Baka**, Algorithmic Solutions for Combinatorial Problems in Resource Management of Manufacturing Environments
140. **Jari Kyngäs**, Solving Challenging Real-World Scheduling Problems
141. **Arho Suominen**, Notes on Emerging Technologies
142. **József Mezei**, A Quantitative View on Fuzzy Numbers
143. **Marta Olszewska**, On the Impact of Rigorous Approaches on the Quality of Development
144. **Antti Airola**, Kernel-Based Ranking: Methods for Learning and Performace Estimation
145. **Aleksi Saarela**, Word Equations and Related Topics: Independence, Decidability and Characterizations
146. **Lasse Bergroth**, Kahden merkkijonon pisimmän yhteisen alijonon ongelma ja sen ratkaiseminen
147. **Thomas Canhao Xu**, Hardware/Software Co-Design for Multicore Architectures
148. **Tuomas Mäkilä**, Software Development Process Modeling – Developers Perspective to Contemporary Modeling Techniques
149. **Shahrokh Nikou**, Opening the Black-Box of IT Artifacts: Looking into Mobile Service Characteristics and Individual Perception
150. **Alessandro Buoni**, Fraud Detection in the Banking Sector: A Multi-Agent Approach
151. **Mats Neovius**, Trustworthy Context Dependency in Ubiquitous Systems
152. **Fredrik Degerlund**, Scheduling of Guarded Command Based Models
153. **Amir-Mohammad Rahmani-Sane**, Exploration and Design of Power-Efficient Networked Many-Core Systems
154. **Ville Rantala**, On Dynamic Monitoring Methods for Networks-on-Chip
155. **Mikko Pelto**, On Identifying and Locating-Dominating Codes in the Infinite King Grid
156. **Anton Tarasyuk**, Formal Development and Quantitative Verification of Dependable Systems
157. **Muhammad Mohsin Saleemi**, Towards Combining Interactive Mobile TV and Smart Spaces: Architectures, Tools and Application Development
158. **Tommi J. M. Lehtinen**, Numbers and Languages
159. **Peter Sarlin**, Mapping Financial Stability
160. **Alexander Wei Yin**, On Energy Efficient Computing Platforms
161. **Mikołaj Olszewski**, Scaling Up Stepwise Feature Introduction to Construction of Large Software Systems
162. **Maryam Kamali**, Reusable Formal Architectures for Networked Systems
163. **Zhiyuan Yao**, Visual Customer Segmentation and Behavior Analysis – A SOM-Based Approach
164. **Timo Jolivet**, Combinatorics of Pisot Substitutions
165. **Rajeev Kumar Kanth**, Analysis and Life Cycle Assessment of Printed Antennas for Sustainable Wireless Systems
166. **Khalid Latif**, Design Space Exploration for MPSoC Architectures

167. **Bo Yang**, Towards Optimal Application Mapping for Energy-Efficient Many-Core Platforms
168. **Ali Hanzala Khan**, Consistency of UML Based Designs Using Ontology Reasoners
169. **Sonja Leskinen**, m-Equine: IS Support for the Horse Industry
170. **Fareed Ahmed Jokhio**, Video Transcoding in a Distributed Cloud Computing Environment
171. **Moazzam Fareed Niazi**, A Model-Based Development and Verification Framework for Distributed System-on-Chip Architecture
172. **Mari Huova**, Combinatorics on Words: New Aspects on Avoidability, Defect Effect, Equations and Palindromes
173. **Ville Timonen**, Scalable Algorithms for Height Field Illumination
174. **Henri Korvela**, Virtual Communities – A Virtual Treasure Trove for End-User Developers
175. **Kameswar Rao Vaddina**, Thermal-Aware Networked Many-Core Systems
176. **Janne Lahtiranta**, New and Emerging Challenges of the ICT-Mediated Health and Well-Being Services
177. **Irum Rauf**, Design and Validation of Stateful Composite RESTful Web Services
178. **Jari Björne**, Biomedical Event Extraction with Machine Learning
179. **Katri Haverinen**, Natural Language Processing Resources for Finnish: Corpus Development in the General and Clinical Domains
180. **Ville Salo**, Subshifts with Simple Cellular Automata
181. **Johan Ersfolk**, Scheduling Dynamic Dataflow Graphs
182. **Hongyan Liu**, On Advancing Business Intelligence in the Electricity Retail Market
183. **Adnan Ashraf**, Cost-Efficient Virtual Machine Management: Provisioning, Admission Control, and Consolidation
184. **Muhammad Nazrul Islam**, Design and Evaluation of Web Interface Signs to Improve Web Usability: A Semiotic Framework
185. **Johannes Tuikkala**, Algorithmic Techniques in Gene Expression Processing: From Imputation to Visualization
186. **Natalia Díaz Rodríguez**, Semantic and Fuzzy Modelling for Human Behaviour Recognition in Smart Spaces. A Case Study on Ambient Assisted Living
187. **Mikko Pänkäälä**, Potential and Challenges of Analog Reconfigurable Computation in Modern and Future CMOS
188. **Sami Hyrynsalmi**, Letters from the War of Ecosystems – An Analysis of Independent Software Vendors in Mobile Application Marketplaces
189. **Seppo Pulkkinen**, Efficient Optimization Algorithms for Nonlinear Data Analysis
190. **Sami Pyöttiälä**, Optimization and Measuring Techniques for Collect-and-Place Machines in Printed Circuit Board Industry
191. **Syed Mohammad Asad Hassan Jafri**, Virtual Runtime Application Partitions for Resource Management in Massively Parallel Architectures
192. **Toni Ernvall**, On Distributed Storage Codes
193. **Yuliya Prokhorova**, Rigorous Development of Safety-Critical Systems
194. **Olli Lahdenoja**, Local Binary Patterns in Focal-Plane Processing – Analysis and Applications
195. **Annika H. Holmbom**, Visual Analytics for Behavioral and Niche Market Segmentation
196. **Sergey Ostroumov**, Agent-Based Management System for Many-Core Platforms: Rigorous Design and Efficient Implementation
197. **Espen Suenson**, How Computer Programmers Work – Understanding Software Development in Practise
198. **Tuomas Poikela**, Readout Architectures for Hybrid Pixel Detector Readout Chips
199. **Bogdan Iancu**, Quantitative Refinement of Reaction-Based Biomodels
200. **Ilkka Törmä**, Structural and Computational Existence Results for Multidimensional Subshifts
201. **Sebastian Okser**, Scalable Feature Selection Applications for Genome-Wide Association Studies of Complex Diseases
202. **Fredrik Abbors**, Model-Based Testing of Software Systems: Functionality and Performance
203. **Inna Pereverzeva**, Formal Development of Resilient Distributed Systems
204. **Mikhail Barash**, Defining Contexts in Context-Free Grammars
205. **Sepinoud Azimi**, Computational Models for and from Biology: Simple Gene Assembly and Reaction Systems
206. **Petter Sandvik**, Formal Modelling for Digital Media Distribution

207. **Jongyun Moon**, Hydrogen Sensor Application of Anodic Titanium Oxide Nanostructures
208. **Simon Holmbacka**, Energy Aware Software for Many-Core Systems
209. **Charalampos Zinoviadis**, Hierarchy and Expansiveness in Two-Dimensional Subshifts of Finite Type
210. **Mika Murtojärvi**, Efficient Algorithms for Coastal Geographic Problems
211. **Sami Mäkelä**, Cohesion Metrics for Improving Software Quality
212. **Eyal Eshet**, Examining Human-Centered Design Practice in the Mobile Apps Era
213. **Jetro Vesti**, Rich Words and Balanced Words
214. **Jarkko Peltomäki**, Privileged Words and Sturmian Words
215. **Fahimeh Farahnakian**, Energy and Performance Management of Virtual Machines: Provisioning, Placement and Consolidation
216. **Diana-Elena Gratie**, Refinement of Biomodels Using Petri Nets
217. **Harri Merisaari**, Algorithmic Analysis Techniques for Molecular Imaging
218. **Stefan Grönroos**, Efficient and Low-Cost Software Defined Radio on Commodity Hardware
219. **Noora Nieminen**, Garbling Schemes and Applications
220. **Ville Taajamaa**, O-CDIO: Engineering Education Framework with Embedded Design Thinking Methods
221. **Johannes Holvitie**, Technical Debt in Software Development – Examining Premises and Overcoming Implementation for Efficient Management
222. **Tewodros Deneke**, Proactive Management of Video Transcoding Services
223. **Kashif Javed**, Model-Driven Development and Verification of Fault Tolerant Systems
224. **Pekka Naula**, Sparse Predictive Modeling – A Cost-Effective Perspective
225. **Antti Hakkala**, On Security and Privacy for Networked Information Society – Observations and Solutions for Security Engineering and Trust Building in Advanced Societal Processes
226. **Anne-Maarit Majanoja**, Selective Outsourcing in Global IT Services – Operational Level Challenges and Opportunities
227. **Samuel Rönnqvist**, Knowledge-Lean Text Mining
228. **Mohammad-Hashem Hahgbayan**, Energy-Efficient and Reliable Computing in Dark Silicon Era
229. **Charmi Panchal**, Qualitative Methods for Modeling Biochemical Systems and Datasets: The Logicome and the Reaction Systems Approaches
230. **Erkki Kaila**, Utilizing Educational Technology in Computer Science and Programming Courses: Theory and Practice
231. **Fredrik Robertsén**, The Lattice Boltzmann Method, a Petaflop and Beyond
232. **Jonne Pohjankukka**, Machine Learning Approaches for Natural Resource Data
233. **Paavo Nevalainen**, Geometric Data Understanding: Deriving Case-Specific Features
234. **Michal Szabados**, An Algebraic Approach to Nivat's Conjecture
235. **Tuan Nguyen Gia**, Design for Energy-Efficient and Reliable Fog-Assisted Healthcare IoT Systems
236. **Anil Kanduri**, Adaptive Knobs for Resource Efficient Computing
237. **Veronika Suni**, Computational Methods and Tools for Protein Phosphorylation Analysis
238. **Behailu Negash**, Interoperating Networked Embedded Systems to Compose the Web of Things
239. **Kalle Rindell**, Development of Secure Software: Rationale, Standards and Practices
240. **Jurka Rahikkala**, On Top Management Support for Software Cost Estimation
241. **Markus A. Whiteland**, On the k-Abelian Equivalence Relation of Finite Words
242. **Mojgan Kamali**, Formal Analysis of Network Routing Protocols
243. **Jesús Carabaño Bravo**, A Compiler Approach to Map Algebra for Raster Spatial Modeling
244. **Amin Majd**, Distributed and Lightweight Meta-heuristic Optimization Method for Complex Problems
245. **Ali Farooq**, In Quest of Information Security in Higher Education Institutions: Security Awareness, Concerns, and Behaviour of Students
246. **Juho Heimonen**, Knowledge Representation and Text Mining in Biomedical, Healthcare, and Political Domains

247. **Sanaz Rahimi Moosavi**, Towards End-to-End Security in Internet of Things based Healthcare
248. **Mingzhe Jiang**, Automatic Pain Assessment by Learning from Multiple Biopotentials
249. **Johan Kopra**, Cellular Automata with Complicated Dynamics
250. **Iman Azimi**, Personalized Data Analytics for Internet-of-Things-based Health Monitoring
251. **Jaakko Helminen**, Systems Action Design Research: Delineation of an Application to Develop Hybrid Local Climate Services
252. **Aung Pyae**, The Use of Digital Games to Enhance the Physical Exercise Activity of the Elderly: A Case of Finland
253. **Woubishet Zewdu Taffese**, Data-Driven Method for Enhanced Corrosion Assessment of Reinforced Concrete Structures
254. **Etienne Moutot**, Around the Domino Problem – Combinatorial Structures and Algebraic Tools
255. **Joonatan Jalonen**, On Some One-Sided Dynamics of Cellular Automata
256. **Outi Montonen**, On Multiobjective Optimization from the Nonsmooth Perspective
257. **Tuomo Lehtilä**, On Location, Domination and Information Retrieval
258. **Shohreh Hosseinzadeh**, Security and Trust in Cloud Computing and IoT through Applying Obfuscation, Diversification, and Trusted Computing Technologies
259. **Ashok Kumar Veerasamy**, Predictive Models As Early Warning Systems For Student Academic Performance In Introductory Programming

Turku Centre *for* Computer Science

http://www.tucs.fi
tucs@abo.fi

University of Turku
Faculty of Science and Engineering
- Department of Future Technologies
- Department of Mathematics and Statistics

Turku School of Economics
- Institute of Information Systems Science

Åbo Akademi University
Faculty of Science and Engineering
- Computer Engineering
- Computer Science

Faculty of Social Sciences, Business and Economics
- Information Systems

ISBN 978-952-12-4014-0
ISSN 1239-1883

Ashok Kumar Veerasamy

Predictive Models As Early Warning Systems For Student Academic Performance

Ashok Kumar Veerasamy

Predictive Models As Early Warning Systems For Student Academic Performance

Ashok Kumar Veerasamy

Predictive Models As Early Warning Systems For Student Academic Performance