Automatic classification of students in online courses using machine learning techniques

David Monllaó Olivé
July 2019

Department of Computer Science and Software Engineering
School of Physics, Mathematics and Computing
Thesis Declaration

I, David Monllaó Olivé, certify that:

This thesis has been substantially accomplished during enrolment in the degree.

This thesis does not contain material which has been accepted for the award of any other degree or diploma in my name, in any university or other tertiary institution.

No part of this work will, in the future, be used in a submission in my name, for any other degree or diploma in any university or other tertiary institution without the prior approval of The University of Western Australia and where applicable, any partner institution responsible for the joint-award of this degree.

This thesis does not contain any material previously published or written by another person, except where due reference has been made in the text.

The work(s) are not in any way a violation or infringement of any copyright, trademark, patent, or other rights whatsoever of any person.

The work described in this thesis was funded by the Australian government and The University of Western Australia through the Research Training Program (RTP) and by Moodle Pty Ltd from September 2016 to November 2017.

This thesis contains published work and/or work prepared for publication, some of which has been co-authored.

Signature:

Date: 15/07/20
Abstract

From the students to the teacher and the institution that offers a course, they are all interested in having students pass the course in which they are enrolled. Although being engaged in a course does not guarantee that a student will successfully pass it or that they will achieve the goals that motivate them to enrol in it, being able to detect and monitor the level of engagement would allow the lecturer to take early intervention. In traditional face-to-face education, it may be easy for the teacher to detect which students are not engaged or when they have difficulties. Nowadays, ubiquitous learning is common, and as its popularity grows, new challenges also arise. For instance, it is more difficult for teachers to detect over the Internet who are not engaged and who need more encouragement to continue or to work harder. Fortunately, even though face-to-face interaction is absent in online courses, students’ accesses to course materials in online learning environments like Moodle from different computing devices are stored in the activity logs, which can be analysed. Indeed, indicators ranging from the number of clicks and forum posts to their level of social interactions and feedback reflection that can be extracted from activity logs all capture useful information about the level of engagement of each student and of the whole class. These indicators also help the teacher to identify students who are at risk of dropping out of the course, that is, students who stop participating further course activities at a particular point.

The primary focus of this thesis is to develop supervised learning algorithms to automatically classify students in a course, taking as input the indicators mentioned above. The algorithms learn from past courses where students failing a course or dropping out of a course are known and are used as training labels. The trained algorithms are then applied to ongoing courses to predict students who are at risk of not completing the course up to a certain time in a semester, for instance, two days before the due date of an assignment, half-way through the semester, etc, so that the lecturer can be alerted for early intervention. The supervised learning algorithms implemented in the thesis also take into account the context of the course in the list of indicators, as courses often have different structure, how activities are assessed, and whether the course is entirely or only partially online can all affect the prediction results. The thesis also focuses on the portability (also referred to as transferability) of the prediction models; that is, the ability of a model to predict accurately in an educational context that is different to the context used to train it. Various machine learning algorithms have been implemented and compared in the research work, including the logistic classifier and the multi-level perceptron classifier. Multiple neural network architectures which mix students’ activities and context have also been incorporated into these classifiers. The accuracy of the prediction models using data on new courses and data from new sites has been shown to be in the 71.30-83.09% range. This level of accuracy is considered to be high, given the complexity and variability of the data. The conducted research work in this thesis has led to the development of a supervised learning framework which can be used
for future evaluation of different prediction models. This framework is now part of the Moodle LMS and is available for free for the Moodle user community.
# Table of contents

Thesis Declaration ................................. i

Abstract ........................................... ii

Table of contents ................................... iv

Acknowledgments ..................................... vii

Authorship declaration: Co-authored publications ........ viii

1 Introduction ....................................... 1

2 Literature review ................................... 3
  2.1 Learning Analytics & Educational Data Mining ................................. 3
      Students at risk ................................................. 3
  2.2 Moodle .................................................. 4
  2.3 Supervised Learning .......................................... 6
  2.4 Predictive model's portability ...................................... 7

3 A Supervised Learning framework for Learning Management Systems ........ 9
  3.1 Introduction ............................................. 9
  3.2 Framework Architecture ....................................... 10
      Supervised Learning Abstraction ............................... 11
      Prediction model definition ................................... 14
      CSV file preparation ......................................... 14
      Machine Learning algorithms .................................. 15
      Testing mode ............................................... 17
      Production mode ........................................... 18
  3.3 Use case ............................................... 18
  3.4 Conclusions ............................................ 19
  3.5 Future work ............................................ 20

4 A Supervised Learning Framework: Using assessment to identify students at risk of dropping out of a MOOC ........................................ 22
  4.1 Introduction ............................................ 22
5 A Quest for a one-size-fits-all Neural Network: Early Prediction of Students At Risk in Online Courses

5.1 Introduction ............................................................ 33
Contribution .............................................................. 34
5.2 Datasets ................................................................. 35
Labels ................................................................. 35
Data cleansing ........................................................... 36
5.3 Input features .......................................................... 36
Students’ activity (SA) .................................................... 37
Course and assignment information (CAI) .................................. 39
Peers activity (PA) ......................................................... 41
5.4 Algorithms ............................................................. 42
Overfitting ............................................................... 42
Studied neural networks ................................................... 43
5.5 Evaluation ............................................................. 47
5.6 Results ................................................................. 49
5.7 Conclusions and future work ....................................... 50

6 Conclusions ............................................................... 53
6.1 Future work ............................................................ 54

A Indicators for assessment ................................................. 57
A.1 Positive ratings in database entries ................................. 57
A.2 Positive ratings in forum posts ...................................... 59
A.3 The student attempted at least one quiz ............................ 60
A.4 Percentage of quizzes attempted in the course ..................... 62
Acknowledgments

First of all, I would like to thank my wife Carmen, my daughter Laia and my son Fidel for their patience and support. Carmen helped me deal with the workload a research project requires. To play and to have fun with Laia and Fidel in daily basis helped me relativise the importance of each milestone of the project and helped me to balance work and family life. I would also like to thank my parents for their support throughout my life and my family-in-law as well.

I would also like to express my sincere gratitude to my supervisors: To Du Q. Huynh, my principal supervisor, for having endless patience with me, for all the reviews and feedback she provided and for being always kind and positive. To Mark Reynolds, my co-supervisor for all the tips and feedback he provided. To Martin Dougiamas, my external co-supervisor, for his support and for the constructive discussions about the topic we had during the early stages of the research. Thanks as well to Danyon Wiese and Elizabeth Dalton, my team mates at Moodle Pty Ltd., for the endless conversations we had about analytics and for all the feedback they provided. I would also like to thank all Moodle Pty Ltd staff that participated on the development of the analytics API for Moodle. Thanks to all of them for making of this research project a positive and fun experience.

The Administration staff of the Department of Computer Science and Software Engineering helped me organize meetings with my supervisors and made possible my assistance to conferences. I would like to thank Rosie Kriskans and Charise Baker. Thanks to the Graduate Research School as well, whose work may not be that visible outside of UWA but is equally important.

In the end, I would like to thanks the Australian Government and The University of Western Australia for sponsoring this research, and Moodle Pty Ltd. for allowing me to apply this research project to real world products that will improve people’s life. This research is supported by an Australian Government Research Training Program (RTP) Scholarship.
Authorship declaration: Co-authored publications

This thesis contains work that has been published.

Details of the work:

Location in thesis: Chapter 3
Student’s contribution to the paper: 85%
Co-authors’ signatures:

Du Q. Huynh: Date: 11/07/20

Mark Reynolds: Date: 11/07/20

Martin Dougias: Date: 11/07/20

Damyon Wiese: Date: 11/07/2019
Details of the work:

Location in thesis: Chapter 4
Student’s contribution to the paper: 85%

Co-authors’ signatures:

Du Q. Huynh:
Date: 11/07/20

Mark Reynolds:
Date: 11/07/20

Martin Dougiamas:
Date: 11/07/20

Damyon Wiese:
Date: 11/07/20
Details of the work:
https://doi.org/10.1109/TLT.2019.2911068

Location in thesis: Chapter 5
Student’s contribution to the paper: 85%
Co-authors’ signature:

Du Q. Huynh:
Date: 11/07/20

Mark Reynolds:
Date: 11/07/20

Martin Dougiamas:
Date: 11/07/20

Damyon Wiese:
Date: 11/07/20

Student signature:
Date: 11/07/20

I, Du Q. Huynh, certify that the student’s statement regarding his contribution to each paper listed above is correct.

Coordinating supervisor signature:
Date: 11/07/2019
Chapter 1

Introduction

Education is part of everyone’s life. The technological advances in our society made Distance Education [45] popular. It is common that, even in traditional face-to-face settings, students can also access their materials online. In some cases teaching is fully online. Students’ retention and students’ performance are obvious subjects of interest in Distance Education settings. These subjects are extensively studied by disciplines related to Educational Technology [73] like Learning Analytics or Educational Data Mining, where a large list of variables have been identified as predictors of students at risk of dropping out or students with low performance. The reproducibility of previous studies is challenging [26] as the datasets used for the studies are not released publicly to protect students’ privacy. The portability (also referred to as transferability) of predictive models developed as part of previous studies to other courses or educational settings is therefore also challenging and is one of the main areas of concern [16, 29].

Learning Management Systems’s (LMS) role in the Distance Education landscape is to deliver and manage users, courses and their contents. Despite LMSs usually have a rich set of features like tools to submit assignments and quizzes or integration with external repositories of files, their most popular use is as a repository of static files and as a communication tool between students and teachers. Moodle\(^1\) is an open source educational platform and it is a popular LMS nowadays. All the activity generated by Moodle users is stored in Moodle’s activity log. These activity logs are a good source of information that allows us to predict which students are likely to be at risk of dropping out or having difficulties. These tasks can be formulated as classification problems. Machine Learning is used to perform a task without being explicitly programmed to do so. Classification is one of these tasks. Machine Learning techniques are successfully used in different industries, including education.

This research project has been supported by Moodle Pty Ltd. the company behind the Moodle open source project, and it sets the basis for Moodle’s analytics engine\(^2\). Therefore, this research project aims to produce real-world products for teachers and students. Formal sciences like Computer Science or Machine Learning, applied sciences like Educational Technology (Distance education more specifically) and Social Sciences like Educational Psychology have been considered in this research project, following Moodle’s pedagogical principles.

This thesis is structured as a series of papers. The first paper describes the software framework

\(^1\)https://moodle.org
\(^2\)https://docs.moodle.org/35/en/Analytics
designed to build prediction models using Moodle data and it is presented in Chapter 3 "A Supervised Learning framework for Learning Management Systems". A case study of the software framework is also presented in Chapter 3. The second paper, "A Supervised Learning Framework: Using assessment to identify students at risk of dropping out of a MOOC", is an extended version of the paper presented in Chapter 3 and it describes a different predictive model, also built using the software framework presented in Chapter 3. The third paper, "A Quest for a one-size-fits-all Neural Network: Early Prediction of Students At Risk in Online Courses", tackles the portability of predictive models throughout different educational contexts.
Chapter 2

Literature review

2.1 Learning Analytics & Educational Data Mining

There are multiple Learning Analytics (LA) definitions. A popular definition from the Call for Papers of the 1st International Conference on Learning Analytics & Knowledge (LAK 2011) is that Learning Analytics is "the measurement, collection, analysis and reporting of data about learners and their contexts, for purposes of understanding and optimizing learning and the environments in which it occurs". The term Learning Analytics (LA) includes different disciplines related to education and technology. There is no agreement on a single and clear definition. Different authors have extended the concept of LA to a theoretical framework composed of different dimensions [33] [13]. Educational Data Mining (EDM) is a related field described in The Educational Data Mining website\(^1\) as "an emerging discipline, concerned with developing methods for exploring the unique and increasingly large-scale data that come from educational settings and using those methods to better understand students, and the settings which they learn in". Educational Data Mining has been compared with LA and the main difference that has been identified is that the former contains Academic Analytics whereas the latter does not [80]. The fact that EDM’s initial approach to problems came from a computer science perspective and LA from an educational perspective is another discrepancy. A survey on recent Educational Data Mining methods and applications is available in [5]. Other relatively recent surveys are [7] and [76]. Systematic reviews of both fields can be found in [14, 68, 70]. Machine Learning techniques have been used in EDM and LA [66] [38] [12] [83] with good results from algorithms such as Neural Networks [40] [12] and Support Vector Machines [9].

Students at risk

Two of the main areas of interest of EDM and LA are students’ retention and students’ performance. Several studies in journals that are related to LA, EDM and e-learning are focused on students’ retention and on identifying predictor variables of students’ success, understanding success as completing a course or successfully passing the course. These areas of interest can be considered traditional topics in LA and EDM fields. A highly cited paper in the students retention literature is Tinto’s [82], where a theoretical framework for understanding student behaviour is presented. The conclusions extracted

\(^1\)http://educationaldatamining.org/
from Tinto’s research are closely related to the social constructivism principles [21] in which Moodle is based. According to Tinto’s paper [82], the model suggests that once students are admitted into a degree or into a course the social factors become more important than their background. Therefore, the social integration of the student is one of the main aspects to observe when predicting which students are likely to drop out of a course or a field of study.

Combining multiple sources of data for the prediction of students at risk is common in the LA and EDM literature. An early students’ performance study using Moodle data to predict the final mark of students is [77], where the authors used several classification algorithms to predict students’ performance. A prediction model for early warning of students at risk of drop out was evaluated in [58] using a wide range of variables as predictors, including demographic data, previous grades, and reporting high accuracies. Weekly predictive models are built for a course offered multiple times in [34], the study also reports on the effectiveness of early intervention in comparison with interventions during the last weeks of the course. Coldwell et al [15] added demographic data to the activity generated by the students in the learning management system. The study found correlation between demographic data like gender or nationality, and performance. Another example where demographic data and activity data is combined is the OU Analyse project [51]. The tools developed as part of the project automatically generate dashboards for teachers.

Chaplot et al [12] explored student abandonment in a MOOC using Neural Networks and sentiment analysis. The content of different MOOCS and interactions between students and teachers were studied in [38] to determine how much they affect abandonment. Students’ performance has also been studied [83], this paper is focused on making predictions more interpretable by teachers and on representing the performance information in a comprehensible way. An example of recent studies is [8], where the authors used different classification algorithms to predict academic failure. Other studied prediction models include assignment submissions [24] or students’ learning style classification [1, 39].

While input variables based on demographic data have been part of some studies [42, 58, 66], variables based on the activity generated by the students in the LMS seem to be the target of a larger number of studies. Variables like the number of content pages viewed, the number of original posts or variables based on the hours spent on the course were studied as early as in 2005 [63]. Other studies expanded the list of variables to specific LMS activities like Assignments, Quizzes, Forums or Wikis [84]. Predictor variables based on other LMS features like the messaging system have also been analysed [57]. There is a high degree of variability among different studies and it is difficult to extract reliable conclusions. The predictive power of variables based on LMS activities depends on the course and the different activity types each course uses.

2.2 Moodle

A Learning Management System (LMS) is a popular educational setting. Moodle is an open source educational platform. The first prototypes of Moodle [22] were released in 1999 by Martin Dougiamas as part of his PhD at Curtin University. A second publication together with Peter Charles Taylor [23] used Moodle for a postgraduate course about constructivism. The design and development of Moodle is guided by a “social constructionist pedagogy”. This statement is quoted from https://docs.moodle.org/35/en/Philosophy. The constructivism learning theory [35] is based on the idea
that humans construct knowledge from their experiences. Social constructivism [21] adds the social interactions with our peers to the equation. The constructionist learning theory [69], which is based on the constructivism learning theory, states that humans are more effective at acquiring knowledge when constructing something. Therefore, most Moodle activity types are designed according to these learning theories. Assessment in Moodle varies widely depending on the Moodle activity being assessed. Quizzes can be automatically assessed and feedback can be automatically provided to students depending on their answers. Assignments are usually manually assessed by teachers. So teachers can provide custom feedback to students’ submissions and can also permit specific students to resubmit. Lessons in Moodle are adaptive activities that include content pages and question pages, students are assessed automatically based on their replies to the questions. Moodle also offers other assessment types that require more interaction between students, e.g., students assess other students’ submissions in workshops, or students rate other students’ contribution to an activity in ratings. In the latter example, the contributions can be forum posts, glossary entries or database activity entries. All the assessments are recorded in the Moodle grade book, which provides different reports and information summaries to be generated for teachers. The back-end system of Moodle can be accessed via a web front-end and through a mobile application. When an online course is managed by Moodle, the learning activities of students are logged and stored into the back-end database, which contains the detailed activity log.

By default, Moodle only provides basic reporting capabilities. Its reporting tools allow users (mainly teachers) to access course activity logs, to cluster data by students or by activity, and to generate graphs for aggregated data. Analytics tools to extract insights from activity logs and for visualization are not available in the Moodle LMS; however, a wide range of plugins compatible to Moodle can be separately and easily installed. A few typical analytics tools for Moodle based on proprietary software are:

- X-Ray Learning Analytics, which provides predictive analytics;
- Intelliboard, which is a user-friendly reporting tool;
- Analytics Graphs, which provides graphs to illustrate students’ profiles;
- Analytics local plugin, which provides site analytics based on the Matomo (formerly known as Piwik) analytics platform.

These analytics tools can be found in the Moodle’s plugins database: https://moodle.org/plugins/.

There are two Educational Data Mining tools for Moodle worth mentioning: CVLA [24] which features an assignment submission prediction model and MDM, [56] an Educational Data Mining open source tool that helps to ease the whole knowledge discovery process. Some of these tools are open sourced and some of them have predictive analytics capabilities. However, none of them provides a complete framework to manage the entire cycle from hypothesis testing to actionable insights for the users, typically, the teachers.

https://docs.moodle.org/35/en/Activities
Moodle’s popularity has grown to reach 144,413,576 users at the time of writing this; with 15 million courses distributed in more than 100,000 Moodle sites around the world. However, there are other LMS with an important share of the market. Blackboard Learn features assessments, assignments and different learning modules. Canvas, developed by Instructure, is a modern LMS that shares functionality with Moodle and Blackboard Learn. It provides most functionalities through LTI external apps that can be used in other LMSs as well. Google Classroom is another popular LMS released in 2014 and developed by Google. It features activities like assignments, quizzes and different grading methods.

2.3 Supervised Learning

Machine Learning is a Computer Science field that comprises a set of techniques for modelling relations in data. Supervised Learning is a subfield in which algorithms learn a function that maps the relation between a dependent variable and a set of independent variables. Regression analysis, and the method of least squares more specifically, is the earliest form of supervised learning algorithms. Linear discriminants can be considered an early classifier. Naive Bayes is another popular classifier; it is based on Bayes’ theorem and it assumes that the features are completely independent and so they correlate separately to the label. Unfortunately, this assumption is also the biggest limitation of these models. Logistic regression is another early classifier, it is a generalized linear model that uses a logistic function. As a regression method, it will try to find the function that, applied to all training set samples, better estimates the dependant variable. Another popular algorithm for classification that has been used for a few decades are Decision trees. Each leaf in the tree represents the combination of all its branches edges (input features) until reaching the root. Random forests are closely related to Decision trees. They try to overcome Decision trees’ overfitting problems by creating multiple Decision trees and combining their results. The Support Vector Machine is one of the most popular supervised learning algorithms nowadays. Support Vector Machines split the hyperplane in parts and classifies samples accordingly.

Artificial neural networks (ANN) are popular Supervised Learning algorithms which simulate the synapses between neurons in the brain. ANN improve their accuracy during training by back propagating the difference of the label of a sample that is calculated by the network versus the real value to the network’s former layer, updating the ANN weights in its way there. ANNs have been applied to different problems like computer vision, where Convolutional Neural Networks (CNN) are generally used; CNNs are inspired by animal visual cortex, they look at the adjacent features of each feature to detect patterns. Recurrent Neural Networks (RNN) and Long Short-term Memory units (LSTM) are used for time-series problems. This type of ANN is used in fields such as natural language processing or speech recognition systems. The input to this type of network is a sequence of vectors instead of a single feature vector. They calculate the output by combining each of the input vectors with the internal state resulting from the previous input vectors. This type of network has also been used to predict student dropout.

3 https://moodle.net/stats/
4 https://www.blackboard.com/learning-management-system
5 https://www.canvaslms.com
Data preprocessing [48] is an important task in machine learning. Data preprocessing techniques involve filling missing values, very important in sparse datasets, or to ensure that no outliers values can affect the predictive model performance. Data cleansing involves analysing the datasets to fix incorrect samples and / or features, ensuring that the dataset samples and features are validated according to the studied problem. Data normalization and standardization are other usual machine learning techniques. They are used to ensure that all features fit in a specific range of values to make machine learning algorithms’ life easier.

2.4  Predictive model’s portability

Portability is a key aspect of any predictive model as it guarantees that the predictive model is able to provide accurate predictions using data from different sources. The data used to train supervised learning algorithms and the data used to obtain predictions can come from the same dataset. Portability though involves using data that comes from different sources. Different courses and different LMSs in this case.

The Open Academic Analytics Initiative (OAAI)\(^6\) aimed at “researching issues related to the scaling up of learning analytics technologies and solutions across all of higher education” [42]. Part of the initiative was related to the portability of predictive models between institutions. Several papers about the initiative research findings were published [41, 42, 43, 53]. These studies report high predictive performance across different institutions data. Later studies [29] argue that the portability of predictive models that do not incorporate instructional conditions may be subject to high variability in results when applying the model to other courses: “threats to the validity of the results may emerge such as overestimation or underestimation of certain predictors”. Other recent studies [16] report similar conclusions.

The portability of predictive models across different courses is getting increasing attention from researchers in Learning Analytics and Educational Data Mining fields. In [31], the authors study two different offerings of two different courses, reporting an accuracy of 60% or more when a model trained using data from one of the courses is applied to the other course. Another example of a recent publication where the portability of the authors’ prediction models is evaluated is [62], the authors report high accuracies and conclude that predictive models can be successfully transferred to different educational context when some context-related conditions, like the fact that courses have the same or similar users, are met.

\(^6\)https://www.educause.edu/
Chapter 3

A Supervised Learning framework for Learning Management Systems

3.1 Introduction

There are multiple Learning Analytics definitions. A popular definition from the Call for Papers of the 1st International Conference on Learning Analytics & Knowledge (LAK 2011) is that Learning Analytics is "the measurement, collection, analysis and reporting of data about learners and their contexts, for purposes of understanding and optimizing learning and the environments in which it occurs". Educational Data Mining is a related field described in The Educational Data Mining website\(^1\) as "an emerging discipline, concerned with developing methods for exploring the unique and increasingly large-scale data that come from educational settings and using those methods to better understand students, and the settings which they learn in". A very popular educational setting is the Learning Management Systems (LMS). Moodle\(^2\) is an open source Learning Management System with more than 130 million users around the world\(^3\). The back-end system of Moodle can be accessed via a web front-end and through a mobile application\(^4\). When an online course is managed by Moodle, the learning activities of students are logged and stored into the back-end database, which contains the detailed activity log. Predictive modelling allows us to model the relation between a target (also known as label or dependent variable) and a set of variables (also referred to as independent variables or features). Predictive modelling can be implemented using Supervised Learning algorithms which learn a function that maps this features-target relation. A large number of EDM and LA research papers are focused on predicting students’ course grades or predicting students that might fail a course.

Although successfully passing a course is important, it is merely the result of a learning process rather than the learning process itself. In order to optimise learning processes, prediction models should also target at the learning processes themselves, in whatever shape they come. Learning processes in an LMS course can be part of an activity or they can be an activity or the course itself. Examples of activities that we can find in an LMS course are: assignments, quizzes or lessons. Examples of

---

\(^1\)http://educationaldatamining.org/
\(^2\)https://moodle.org
\(^3\)https://moodle.net/stats/
\(^4\)https://download.moodle.org/mobile
CHAPTER 3. A SUPERVISED LEARNING FRAMEWORK FOR LEARNING MANAGEMENT SYSTEMS

prediction model targets could be: learners failing a quiz, learners submitting assignments after the due date, or learners not being able to complete a lesson in less than 15 minutes. Examples of prediction models that target at the course itself rather than the learners are: will this course engage learners? Will this forum be useful to the students of this course?

The contribution of this paper: The presented software framework provides a solid foundation for the development of EDM and LA prediction models. The framework simplifies the implementation of new prediction models in online educational contexts. This is an advantage over developing and evaluating prediction models from scratch. By using this framework, EDM and LA researchers can focus on the educational aspects of their research instead of spending their time on evaluating models performance using basic statistical methods manually or tuning Machine Learning algorithms. Although the framework is developed for Moodle, the same design can be applied to other Learning Management Systems as the framework components are just Supervised Learning prediction model abstractions for an educational context. Our second contribution is a developed model for predicting students at risk that can be readily used within the framework. This model is able to accurately predict which students are at risk of abandoning a MOOC before it finishes. This allows educators to take appropriate intervention before the end of the course.

3.2 Framework Architecture

The Supervised Learning framework presented in this paper is part of the Moodle LMS since its 3.4.0 version. A Moodle site prediction model is completely separated from other Moodle sites, so each Moodle site only uses the training data and obtains predictions for samples available in that same site. The framework manages the prediction model’ life cycle and, for any given prediction model, the framework supports two different modes:

- It automatically extracts training and validation data from the finished courses according to the features and target defined for the prediction model. It then trains the prediction model and validates its prediction accuracy. This is referred to as the testing mode.

- It uses all the data from the finished courses as training data and the prediction is done for on-going courses. This is referred to as the production mode.

Moodle is written in PHP\textsuperscript{5}, which is not the best programming language for Machine Learning for multiple reasons. The memory overhead added by each PHP variable and the lack of GPU-acceleration support are two major concerns. Because of these, we separate the framework into two layers:

- The Moodle Analytics API\textsuperscript{6} written in PHP is responsible for generating labelled and unlabelled CSV\textsuperscript{7} files from Moodle’s database contents. The CSV format is chosen because it is a simple text format that is portable on all platforms. For all prediction models, multiple CSV files are generated to capture information about the courses and the students.

\textsuperscript{5}http://php.net
\textsuperscript{6}https://docs.moodle.org/dev/Analytics_API
\textsuperscript{7}http://www.ietf.org/rfc/rfc4180.txt#page-1
3.2. FRAMEWORK ARCHITECTURE

Figure 3.1: A Data flow diagram showing the steps that the framework goes through in the production mode. It includes examples of each component.

- Machine Learning backends are responsible for processing these files. They process the labelled and unlabelled CSV files. These backends can be written in any programming language.

This 2-layer design allows us to speed up the training process as well as to allow researchers with Machine Learning experience to write their own Machine Learning backends if they are not satisfied with the default backends’ performance.

Supervised Learning Abstraction

The framework is an abstraction of a Supervised Learning problem. In this section we list the main elements that compose the framework and how they are mapped to the typical elements needed in a Supervised Learning task.

Each prediction model is composed of a single Analyser, a single Target, a set of Indicators, one Time Splitting Method and a single Machine Learning backend. All these elements are implemented as PHP classes\(^8\) although Machine Learning backends can be written in other programming language. The whole Supervised Learning framework has been designed so that the elements described below are

\(^8\)http://php.net/manual/en/language.oop5.php
CHAPTER 3. A SUPERVISED LEARNING FRAMEWORK FOR LEARNING MANAGEMENT SYSTEMS

Figure 3.2: The class diagram of the framework.

Reusable and extensible independently across different prediction models. The framework elements are described below following the order shown in Figure 3.1. Figure 3.2 shows the class diagram of the implemented system.

Analysers

Each Analyzer is responsible for defining the subject of the model. They select and pass all the Moodle data associated to these subjects to Targets and Indicators (described right below) as shown in Figure 3.1.

The following analysers are included in the framework and can be reused by researchers to create new prediction models:

- **Student enrolments**: The subject of the model are students in a course.
- **Users**: The subject of the model are site users.
- **Courses**: The subject of the model are courses.
3.2. FRAMEWORK ARCHITECTURE

Targets

They are the key element of a prediction model. They represent the labels of a Supervised Learning dataset and define the event of interest. Obviously, Targets depend on Analysers, because Analysers provide the subjects that Targets need for calculating the label. The framework includes an identifying students at risk of Target. Here are a few more examples of Targets in prediction models and their associated subjects:

- **Identifying spammer user**: The subjects of the model are site users.
- **Classifying ineffective course**: The subjects of the model are courses.
- **Assessing difficulties to pass a specific quiz**: The subjects of the model are quizzes.

Indicators

They represent the features in a Supervised Learning problem. Indicators are responsible for performing calculations on Moodle data. They are calculated for each subject using data available in the time period defined by the Time splitting method (described right below). They were designed to avoid normalisation issues so the CSV file features generated from indicators always have values in the range $[-1, 1]$. Indicators are in one of the following categories:

- **Linear**: The indicator values are floating point numbers in the range $[-1, 1]$. An example would be "the weight of quiz activities in a course".
- **Binary**: The indicator values are Boolean values $\in \{0, 1\}$. An example would be "has this student completed all activities in the course?".
- **Discrete**: The indicator values are a closed list of values. The framework one-hot encodes the list of values and generates $N$ features with values $\in \{0, 1\}$. An example would be "How often do this student access the course?", with values "never", "in monthly basis" and "in weekly basis".

The framework includes a set of indicators that can be used in new prediction models. Not all indicators can be included in any model. E.g. student-related indicators could not be calculated if the predictions subjects are courses. Below are some of the indicators included in the framework:

- The number of clicks made by the student in a course. It is implemented as a linear indicator.
- Student posts to forum activities in a course. It is implemented as a linear indicator.
- Was the course accessed after the course end date? It is implemented as a binary indicator.
- Was the course accessed before the course start date? It is implemented as a binary indicator.

Time splitting methods

They define when the framework should generate predictions and the time period that should be used to calculate the indicators. The Moodle LMS includes a few time splitting methods that researchers can use in their prediction models. For example:
• Split the course duration into four parts and generate a prediction at the end of each part.
• Generate a prediction one week before each assignment’s due date and generate a second prediction two days before the assignment due date.

Machine Learning backends

In the testing mode, the Machine Learning backends split the CSV files into training and testing sets and evaluate the prediction accuracy. This process is described in more detail in Sections 3.2 and 3.2 below. In the production mode, Machine Learning backends are trained with finished courses data and return predictions for ongoing courses.

The Moodle LMS includes two Machine Learning backends: A Feed-Forward Neural Network [79] written in Python\(^9\) using the Tensorflow framework\(^10\) and a Logistic Regression [18] classifier written in PHP for the Moodle sites where installing Python is not an option. New Machine Learning backends can be plugged on the framework.

Prediction model definition

Researchers can define prediction models by implementing a *Target* in PHP. Depending on the subject (see Section 3.2) researchers can reuse an existing *Analyser* or they can create a new one. The next step that the researcher should perform is to select the Indicators that would have impact on the *Target* and select the Time splitting method from a list of choices provided by the framework.

CSV file preparation

In this section we describe the process the framework follows to generate a labelled CSV file from the Moodle site’s database. The *Analysers* of the framework iterate through the site database, gathers the model subjects’ data from the different analysable elements available in the system. The features in the CSV file are added according to the prediction model *Indicators* and *Time splitting method*. The interaction between the *Analyser* and the *Indicators* has been designed so that the framework is able to automatically add extra features. This allows us to perform feature engineering tasks while still having access to the LMS data. For each *Linear* indicator the framework automatically adds an extra feature whose value is the mean of all samples in that analysable element. The number of features in the resulting CSV file is determined by the following formula:

\[
NF = (NL \times 2) + NB + \sum_{i=1}^{ND} NV_i + TP, \tag{3.1}
\]

where \(NF\) is the number of features, \(NL\) the number of linear indicators, \(NB\) the number of binary indicators, superscript \(ND\) is the number of discrete indicators, \(NV\) the number of values in the \(i^{th}\) discrete indicator and \(TP\) the number of predictions generated by the time splitting method.

The values for *label* are also calculated according to the definition of the *Target*. The framework supports binary classification, multi-class classification and regression, but the Machine Learning

\(^9\)https://www.python.org/
\(^10\)https://www.tensorflow.org/
backends included in Moodle do not yet support multi-class classification or regression, so only binary classifications are fully supported at this stage.

**Machine Learning algorithms**

The CSV files described in Section 3.2 above are consumed by Machine Learning algorithms. As mentioned in Section 3.2, the Machine Learning backends of the framework include two classifiers. These classifiers are described in detail in the following subsections.

**Logistic Regression**

The PHP Machine Learning backend uses a Logistic Regression binary classifier \[18\] to perform its predictions. In Logistic regression, a Logistic function \( h_\theta(x) = \frac{1}{1 + \exp(-\theta^T x)} \) is applied to the feature vector \( x \) to produce a value in the range \((0, 1)\). The parameter \( \theta \in \mathbb{R}^n \) is a vector of weights that need to be learned. The function behaves like a thresholding function with a soft boundary. Because of the range the output value \( h_\theta(x) \) can be interpreted as the probability whether \( x \) belongs to the target class in a two-class classification problem. Logistic Regression tries to find the best fitting model for the relation between features and their labels by optimising the following cross-entropy cost function:

\[
J(\theta) = -\frac{1}{m} \sum_{i=1}^{m} y_i \log(h_\theta(x_i)) + (1 - y_i) \log(1 - h_\theta(x_i)),
\]

where \( m \) is the total number of samples, \( x_i \) denotes the \( i \)th sample and \( y_i \) denotes the corresponding ground truth label. The cost function gives the error between a set of weights where all samples’ fit perfectly and the labels predicted by the algorithm. The parameter \( \theta \) is updated according to the gradients of the cost function. Gradient Descent \[20\] is the common algorithm used to optimise cost functions in Machine Learning. It is iteratively used to update the set of weights \( \theta \) using the following formula:

\[
\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta), \quad \text{for} \quad j = 1, \ldots, n,
\]

where \( \alpha \) is a constant called learning rate which defines how significant \( \theta \) updates are and \( \frac{\partial}{\partial \theta_j} J(\theta) \) is the partial derivative of \( J(\theta) \). Logistic Regression is an effective algorithm for binary classification although some problems would arise when the number of features increases: The algorithm becomes computationally very expensive and it is very easy for the algorithm to over fit the training data. Logistic Regression popularity is decreasing in favour of other Supervised Learning algorithms.

**Feed-Forward Neural Network**

Artificial Neural Networks (ANN) \[78\] mimic biological Neural Networks. They are networks containing relations between units, also called neurons. Neural Networks organise units into layers. The connections between neurons in different layers are represented as a set of weights \( \theta \). The Neural Network improve its accuracy during training by back propagating \[79\] the difference of the calculated label of a sample versus the real value to the network’s previous layers, updating its weights. The connections between different layer neurons become stronger or weaker after the back-propagation process.
Feed-forward Neural Networks contain a first layer of units called input layer, with as many units as features in the input CSV file. They then contain a number of hidden layers \( l \geq 1 \) and a final output layer with as many units as different labels the input CSV file has. The Neural Network included in the framework contains one hidden layer with ten hidden units. The network training process is composed of two different steps: feed-forward and back-propagation.

The feed-forward process computes the predicted labels for a given set of samples by multiplying input features values by the matrices that connect different layer units. The following formulas use a set of samples of size \( N \). In \( z_1 = \theta_1 \cdot x + b \) we multiply the input features matrix for the weights matrix \( \theta_1 \) that connects the input features \( x \) with the first hidden layer units, adding a bias \( b \). \( z_1 \) is multiplied by a non-linear activation function \( g(z) \). Different activation functions can be used, some examples are \( g(z) = \text{sigmoid}(z) \), \( g(z) = \text{tanh}(z) \) or \( g(z) = \text{relu}(z) \). The activated matrix (one vector for each sample) in the following layer is therefore expressed as \( a_1 = g(z_1) \). This calculation is repeated until the output layer is reached. It can be generalised as \( a_l = g(\theta_l \cdot a_{l-1}) \). The softmax function can be used as the output layer activation function as it returns a \( \in [0,1] \) value for each possible label. The softmax formula is expressed as follows:

\[
\text{softmax}_i = \frac{e^{a_{li}}}{\sum_j e^{a_{lj}}},
\]

where \( i \) and \( j \) are each of the possible labels.

We finish the forward pass by calculating the error using the cross-entropy cost function \( J(\theta) \) given in Eq. 3.2.

During back-propagation we minimise the cost function \( J(\theta) \) by updating the weights that connect neurons in different layers. The updated value depends on the partial derivative of the error with respect to each of the network weights in the previous layer \( \partial J(\theta) / \partial \theta_{ij} \) and the learning rate \( \alpha \). Weights are updated using the Delta rule, expressed as follows:

\[
\Delta \theta_l = -\alpha \frac{\partial J(\theta)}{\partial \theta_l}.
\]

The partial derivatives calculations are based on the chain rule, which allows us to compute a derivative as the composition of two or more functions, in our case:

\[
\frac{\partial J(\theta)}{\partial \theta_l} = \frac{\partial J(\theta)}{\partial g(z_l)} \cdot \frac{\partial g(z_l)}{\partial z_l} \cdot \frac{\partial z_l}{\partial \theta_l},
\]

where \( \theta_l \) represents the vector of weights in layer \( l \) and \( g(z) \) the activation function. \( \partial z_l / \partial \theta_l = a_l \), the partial derivative of the activation output is the derivative of the activation function. E.g. Sigmoid function \( g'(z_l) = g(z_l)\cdot(1 - g(z_l)) \):

Finally, \( \partial J(\theta) / \partial g(z_l) \) calculation depends on the value of \( l \) as we need to consider all the layers from \( l \) to the output layer. We calculate deltas \( \delta_l \) for each layer starting from the output layer one, which is \( \delta_{\text{output}} = (\hat{y} - y) \), where \( y \) and \( \hat{y} \) are, respectively, the actual and predicted labels (both are vectors usually). With \( \delta_{\text{output}} \) we can go backwards calculating the previous deltas until the first hidden layer with \( \delta_l = \theta_l^T \delta_{l+1} \cdot g'(z_l) \). The partial derivative of any weight in the network is therefore
3.2. FRAMEWORK ARCHITECTURE

represented as:

\[
\frac{\partial J(\theta)}{\partial \theta_t} = \delta_{t+1} \cdot g(z_t).
\]  

(3.7)

Testing mode

In Section 3.2 we described that the framework generates a labelled CSV file from Moodle site contents based on the prediction model defined by the researcher. In this section we describe how the framework evaluates the defined prediction model using Machine Learning techniques.

The first thing the evaluation process detects are the CSV files with highly skewed classes. The provided Machine Learning backends do not cope well with really unbalanced classes. Even if the Machine Learning backend reports high accuracies the recall or the precision will probably not be high which would lead to a low predictive power. Prediction models with highly skewed classes are not further evaluated in the current framework.

The Machine Learning algorithm is trained with the training dataset and the test dataset is used to calculate the Matthews’ correlation coefficient [59] of the prediction model. The Matthews’ correlation coefficient is a good evaluation metric for binary classification problems because it takes into account both true positives and true negatives [71].

To calculate the Matthews’ correlation coefficient we first fill out the confusion matrix with the predicted results and the test dataset labels. The confusion matrix looks like this:

\[
\begin{bmatrix}
TN & FN \\
FP & TP
\end{bmatrix},
\]

where TP = True positives, FP = False positives, TN = True negatives and FN = False negatives. The Matthews’ correlation coefficient (MCC) formula is given by:

\[
MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP) \times (TP + FN) \times (TN + FP) \times (TN + FN)}}.
\]  

(3.8)

The entire process is automated. So this training and MCC calculation process described above is repeated multiple times and the MCC for each iteration is be recorded. The average MCC and standard deviation is later calculated from all iterations. An average MCC of 0 indicates that the model is no better than a random model, a negative value indicates an inverse relation and a positive value a positive relation. Therefore, the higher the MCC value is, the better the model is at predicting. The standard deviation of all the calculated MCC will be used to detect variations in the coefficients. Small variances in each iteration’s MCC can be expected because the CSV file contents are shuffled before each iteration, but high variances are a good sign that the CSV file is not large enough to guarantee that the evaluation results are reliable.

The average MCC is automatically converted to a score in the range \([0, 100]\) using the following formula:

\[
score = \frac{MCC + 1}{2}.
\]  

(3.9)

The computed score is provided to the researcher as a quality measure for the prediction model.
CHAPTER 3. A SUPERVISED LEARNING FRAMEWORK FOR LEARNING MANAGEMENT SYSTEMS

Production mode

The testing mode is useful for checking the hypotheses of the EDM / LA researchers and to verify existing educational literature theories. Once the researcher (or the site administrator if the researcher do not have enough permissions on the site) switches a model to production mode, these hypotheses and theories into optimised learning processes by generating actionable insights from predictions. This production model switch affects the whole site: Insights are generated for all stakeholders in the site, usually the teachers, in addition to the researchers who switch the framework to work in production mode. E.g. Arts teachers do nothing to receive predictions about their at-risk students. They would receive notifications about at-risk students once the prediction model is globally enabled for production mode.

As part of the Target definition, researchers can specify which predictions returned by the Machine Learning backend are worth observing and which predictions can be ignored. E.g. A model that predicts students at risk is only interested in at-risk students, not in students that progress as expected. Actionable insights can be specified as part of the Target definition as well. These actionable insights are provided as suggested actions to the site user that have access to predictions, usually a teacher in a course. Examples of actionable insights can be to message the at-risk student, to check the student activity log or to ignore the prediction. Figure 3.4 is an example of how insights are presented to teachers.

The state of the Machine Learning algorithms is saved once the training process finishes. The trained algorithm is restored every time Machine Learning backends need to generate predictions for ongoing courses. The CSV file generated by Moodle for ongoing courses is obviously unlabelled.

3.3 Use case

In this section, we report one of the implemented prediction models of the Supervised Learning framework. This prediction model classifies students without activity logs during the last quarter of a course as drop-outs and all other students as not-drop-outs. We implemented a Time splitting method to generate 3 predictions along the course duration. The first one is executed once the first quarter of the course is over, using data from the start of the course. The second one is executed after half course is completed and use activity data from the beginning of the course up to that point. The third prediction is executed after the third quarter of the course, also using all the data available from the start of the course up to that point in time.

A significant amount of Learning Analytics literature is focused on describing online students’ engagement from different educational paradigms [60]. Some of these conceptual frameworks, like Community of Inquiry [28], are very popular among educators. Community of Inquiry started as an exploratory and descriptive framework, recognising later its limitations from an empirical point of view [27] The Indicators we used in the implemented at-risk students model are based on this paradigm. The adaptation of the Community of Inquiry paradigm to Moodle deserves a separate paper. As a brief summary, we implemented linear indicators based on how intensive the student’s interactions with course activities are. Students with no interactions in activities result in negative values and students that interacted with teachers and other peers result in positive values.
3.4. CONCLUSIONS

To test the prediction model we used 8 finished anonymised MOOCs with a total of 46,895 students. Results provided by the framework testing mode are shown in Table 4.1 and Figure 3.3.

Our test results show that the proposed at-risk students model gave an average prediction accuracy of 92.56% using the Neural Network described in Section 3.2 and an average prediction accuracy of 73.30% using the Logistic Regression classifier described in Section 3.2. The Neural Network appear to be better at modelling the relation between the input CSV file features and the label, probably due to the extra set of weights in the hidden layer which should allow the Neural Network to model more complex relations between features.

Once the production mode is enabled, predictions can be generated for ongoing courses. The actionable insights generated by this at-risk students model are shown in Figure 3.4.

3.4 Conclusions

The framework design covers prediction models like at-risk students and allows researchers to evaluate them to determine if they are effective to be used in production. To create a prediction model using the
CHAPTER 3. A SUPERVISED LEARNING FRAMEWORK FOR LEARNING MANAGEMENT SYSTEMS

Figure 3.4: Actionable insights generated by an at-risk students model.

presented framework is an advantage over creating a prediction model from scratch as the framework provides a set of base elements and a number of evaluation metrics to test prediction models accuracy. A prediction model to detect late assignment submissions \(^\text{11}\) has also been used to check that the software framework design is adaptable to different prediction subjects and that the framework indicators can be reused across prediction models. The accuracy and recall of the presented prediction model for predicting at-risk students are good for a production system. The Supervised Learning framework presented in this paper is part of the Moodle LMS from version 3.4.0 onward, but is disabled by default as it requires sufficient computer power to run. Moodle is an open source Learning Management System so the exact number of unregistered users is unknown. Given that there are more than 130 million registered users around the world and given that approximately 50% of more than 100,000 registered Moodle sites use Moodle 3.4.0 or above \(^\text{12}\) (June 2018), it would not be too unrealistic to claim that the framework is used by millions of users.

3.5 Future work

There are multiple ideas worth exploring:

- The framework could be extended to cover Unsupervised Learning using the same 2-layer architecture. The framework has been designed keeping Unsupervised Learning in mind so the

\(^\text{11}\) https://github.com/dmonllao/moodle-local_latesubmissions

\(^\text{12}\) https://moodle.net/stats/
same set of indicators for Supervised Learning could be used for Unsupervised Learning.

- Learning Analytics literature can be reviewed and extra student engagement indicators can be extracted from the literature. These extra indicators can be adapted to the data available in the LMS, be implemented in PHP and added to the at-risk students model. As it is, the testing mode of the framework can be used to check which of these indicators result in accuracy improvements, supporting the results obtained in other research papers. It is important to mention that the Neural Network and the Logistic Regression classifier described in Section 3.2 and included in the Moodle LMS do not try to compete with the most accurate algorithms available nowadays and their performance can be easily surpassed by tuned Neural Networks and other Machine Learning techniques.

- Exploring how cross-validation can improve the adaptability of the Machine Learning algorithms included in the Moodle LMS is another option for the future. Cross-validation processes are useful to adapt Machine Learning algorithms to datasets so the trained algorithms are more accurate. To tune Machine Learning algorithms using cross-validation before training would be a very good option if our training datasets remain static. In our case though, new training data is available in regular basis (e.g. every time a course is finished) so cross-validation can be a double-edged sword.
Chapter 4

A Supervised Learning Framework: Using assessment to identify students at risk of dropping out of a MOOC

4.1 Introduction

MOOCs, or Massive Open Online Courses, facilitate the access to online education. Their use is increasing and online platforms such as Coursera, edX or Udacity\(^1\) have millions of students. The teacher-student ratio in MOOCs is lower than in traditional face-to-face courses, and the relation between the student and the teacher is not as direct as it is in face-to-face learning, therefore new challenges arise. Drop outs, the term drop-out and variants like not-drop-outs refer to students who abandon the course before its completion. Drop out ratios in MOOCs are generally high as reported in [3] and [38]. Therefore, to quickly identify students that require attention is very important for MOOCs.

Assessments in online courses contain a valuable source of information, and the correlation between students at risk of dropping out and variables related to assessment have been studied in a large number of research papers. Typical examples include [2], where a novel analysis to estimate students’ retention in MOOCs is conducted, and [8], where the authors use logistic regression to identify students that will likely drop out of courses. Assessments for learning are used by both the teacher and the students: The teacher uses assessment to identify students’ learning needs and strengths, and adapts the course contents accordingly; the students use assessment to benchmark their activities and performance, as well as to improve their submissions and their understanding of the concepts and topics they are assessed on.

Predictive modelling allows us to identify students at risk in ongoing courses. Prediction models can be implemented using Supervised Learning algorithms which learn a function that maps a set of independent variables (also referred to as features) to a target (also known as label or dependent variable). Creating and evaluating prediction models is a complex task and some of these tasks are common among most prediction models. The MOOCs used in this study are offered as Moodle courses.

Moodle\textsuperscript{2} is an open source learning platform used for e-learning [4] which makes it convenient for developing our Supervised Learning framework. An attempt to tackle the same problem in Moodle is Moodle Data Mining (MDM) [56] which is an Educational Data Mining open source tool targeting at easing the whole knowledge discovery process. However, unlike our framework, MDM does not provide a complete framework that manages the entire cycle from hypothesis testing to generation of actionable insights.

This paper is an extended version of our work published in [61]. The main extension includes the development of a new prediction model for the supervised learning framework.

The contributions of this paper are twofold:

- The first contribution is the presented software framework. It provides a solid foundation for the development of EDM and LA prediction models. The framework simplifies future implementation of new prediction models in online educational contexts and it supports the reusability of their components.

- A case study predictive model for identifying students at risk of dropping out that achieves an accuracy of 88.81% and an F1 score of 0.9337. The variables used to identify students at risk are based on automatically assessed activities.

The paper is organised as follows: The Implementation of the supervised Learning framework is described afterwards in Section 4.2, followed by the Methodology used to evaluate predictive models in Section 4.3. A Case study is described in Section 4.4. This section includes a description of the Dataset and the Results. The Discussion section interprets the significance of this paper in relation to what was already known and the Conclusions are highlighted in Section 4.6. Finally, possible future lines of research are listed in Section 4.7.

4.2 Implementation

The Supervised Learning framework presented in this paper is part of the Moodle LMS from 3.4.0 onward. A prediction model in a Moodle site is completely separated from the same prediction model in another Moodle site. In other words, each Moodle site only uses the training data and obtains prediction samples available in that same site. The framework manages the life cycle of the prediction model and, for any given prediction model, the framework supports two different modes:

- Testing mode: In this mode, the framework automatically extracts training and validation data from the finished courses according to the indicators and target defined for the prediction model. It then trains the prediction model and validates its prediction accuracy.

- Production mode: It uses all the data from the finished courses as training data and the prediction is done for on-going courses.

The framework is separated into two layers:

\textsuperscript{2}https://moodle.org
The Moodle Analytics API, written in PHP is responsible for generating labelled and unlabelled CSV files from Moodle’s database contents. The CSV format is chosen because it is a simple text format that is portable on all platforms. For all prediction models, multiple CSV files are generated to capture information about the courses and the students.

The Machine Learning backends are responsible for processing these files. They process the labelled and unlabelled CSV files. These backends can be written in any programming language.

This 2-layer design allows us to speed up the training process as well as to allow researchers with Machine Learning experience to write their own Machine Learning backends if they are not satisfied with the default backends’ performance.

The site analysis performed by the framework runs through scheduled jobs in the cron and it scales to big Moodle sites. Different parameters can be customised so that the analysis can be split in multiple jobs and the duration of the jobs can be limited.

## Components

Each prediction model is composed of a single Analyser, a single Target, a set of Indicators, one Time splitting method and a single Machine Learning backend. All these elements are implemented as PHP classes although Machine Learning backends can be written in other programming languages. The whole Supervised Learning framework has been designed so that the elements described below are reusable and extensible independently across different prediction models. The framework elements are described below, following the order shown in Figure 3.1.

### Analysers

Analysers are responsible for extracting the list of samples from the Moodle database. They iterate through the database and gather the model samples’ data from different analysable elements available in the system. They finally pass all the Moodle data associated to these samples to Targets and Indicators as shown in Figure 3.1.

The following three Analysers are included in the framework and can be reused to create new prediction models: student enrolments, which provides students in a course as samples; users, which provides users as samples; and courses, which provides courses as samples.

### Targets

They are the key element that a prediction model tries to predict. They represent the labels of a Supervised Learning dataset and define the event of interest. Obviously, Targets depend on Analysers, because Analysers provide the samples that Targets need for calculating the labels. A few examples of Targets in prediction models are: a label (e.g., 0 or 1) that distinguishes whether a student is at risk of dropping out of a course, that identifies a user as a spammer, that classifies a course as ineffective, or that assesses whether a specific quiz is difficult to pass.

---

3http://www.ietf.org/rfc/rfc4180.txt#page-1
4.2. IMPLEMENTATION

Indicators

Indicators are functions that generate the features in a Supervised Learning problem. They are calculated for each sample using the data for the time period defined by the Time splitting method (described below). The values of all indicators are restricted to a certain range of values so that normalization is not required. All indicators are in one of the following categories:

- **Linear.** In this category, the indicator values are floating point numbers in the range $[-1, 1]$, e.g., a floating point number denoting the weight of quiz activities in a course.

- **Binary.** In this category, the indicator values are "true" and "false", therefore $\in \{0, 1\}$, e.g., a 0 value to denote that a student has not completed all activities in the course.

- **Discrete.** (modification to replace the previous item) The indicator values are defined from a list of items. Suppose that the list has $N$ items. The framework would convert them using one-hot representation and generate $N$ features with values $\in \{0, 1\}$. For example, for a column in the CSV file that describes "How often do this student access the course?", the associated indicator could be a one-hot vector of 3 elements long capturing the values "never", "in a monthly basis", and "in a weekly basis".

The framework includes a set of indicators that can be used in future implementation of new prediction models. Examples of indicators are: "the number of clicks made by a student in a course" [16] or "the number of posts submitted by a student to forum activities in a course" [25]. It should be noted that not all indicators are suitable for all models. For example, student-related indicators should not be used if the samples provided by the Analyser are for courses.

Time splitting methods

They define when the framework should generate predictions and the time period that should be used to calculate the indicators. The Moodle LMS includes a few default time splitting methods that the users can use in their prediction models. For example, the user can split the course duration into four parts and request to generate a prediction at the end of each part and/or generate a prediction one week before each assignment’s due date and/or generate a second prediction two days before the assignment due date.

Machine Learning backends

In the testing mode, the Machine Learning backends split the CSV files into training and testing sets, using a 20% of the data for testing, and evaluate the prediction accuracy. In the production mode, the Machine Learning algorithms that are part of the backend are trained with finished courses data. The trained model can be used to make predictions for ongoing courses.

The Moodle LMS includes two default Machine Learning backends: A Feed-Forward Neural Network [79] written in Python using the Tensorflow framework, and a Logistic Regression classifier [18] written in PHP for the Moodle sites where installing Python is not an option. New Machine Learning backends can be added as plug-ins to the framework. Users can also select the prediction model to be used
by each prediction model in the system. The implementation of the Logistic Regression classifier is described in detail in our previous work [61].

4.3 Methodology

Prediction model definition

Researchers can define prediction models by implementing a Target in PHP. Depending on the sample type (see Section “Target”) the framework users can reuse an existing Analyser or they can create a new one. The next step that the researcher should perform is to select the Indicators that would have impact on the Target and select the Time splitting method from a list of choices provided by the framework.

CSV file preparation

In this section, the process carried out by the framework to generate a labelled CSV file from the Moodle site’s database is described. The features in the CSV file are appended according to the Indicators and the Time splitting method. The interaction between the Analyser and the Indicators has been designed so that the framework is able to automatically add extra features. This allows us to perform feature engineering tasks while still having access to the LMS data. For each Linear indicator, the framework automatically adds an extra feature whose value is the mean of all samples in that analysable element. The number of features in the resulting CSV file is determined by the following formula:

$$NF = (NL * 2) + NB + \sum_{i=1}^{ND} NV_i + NP,$$

(4.1)

where NF denotes the number of features, NL the number of linear indicators (multiplied by two for the extra feature added for each linear feature as described in the paragraph above), NB the number of binary indicators, ND the number of discrete indicators which result in multiple features being generated from one-hot encoding, with NV\textsubscript{i} being the number of values in the \textsuperscript{i}th discrete indicator, and finally NP represents the number of predictions generated by the time splitting method.

The labels are calculated according to the definition of the Target. The current version of the framework supports binary classification, multi-class classification and regression, but the Machine Learning backends included in Moodle do not yet support multi-class classification and regression.

Testing mode

This section describes how the framework evaluates the defined prediction model using Machine Learning techniques.

The first step of the evaluation process of the framework is to detect those CSV files that have highly skewed data. The current Machine Learning backends do not cope well with really unbalanced classes. Even if the Machine Learning backend may report high accuracies for these CSV files, their recall and/or the precision are often not high enough to guarantee good predictions.
4.3. METHODOLOGY

The Machine learning algorithm included in the *Machine Learning backend* is trained with the training dataset and the test dataset is used to calculate the *Matthews’ correlation coefficient* [59] of the prediction model. The *Matthews’ correlation coefficient* is a good evaluation metric for binary classification problems because it takes into account both true positives and true negatives [71].

To calculate the *Matthews’ correlation coefficient*, the confusion matrix
\[
\begin{bmatrix}
    TN & FN \\
    FP & TP
\end{bmatrix},
\]
needs to be completed using the predicted results and the ground truth test labels. Here, TP = **True positives**, FP = **False positives**, TN = **True negatives** and FN = **False negatives**. These values are then used to compute the *Matthews’ correlation coefficient* (MCC):
\[
MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP) \times (TP + FN) \times (TN + FP) \times (TN + FN)}}.
\]

An average MCC of 0 indicates that the model is no better than a random model, a negative value indicates an inverse relation and a positive value a positive relation. Therefore, the higher the MCC value is, the better the model is at predicting.

The entire process described above is automated. So this training and MCC calculation process is repeated multiple times and the MCC for each iteration is be recorded. The average and standard deviation MCC are later calculated from all the iterations. The standard deviation of all the calculated MCC will be used to identify variations in the coefficients. Small variances in each iteration’s MCC can be expected because the CSV file contents are shuffled before each iteration, but high variances are a good sign that the CSV file is not large enough to guarantee that the evaluation results are reliable.

The *Matthews’ correlation coefficient* (MMC) is used to measure the correlation between the drop-out label and each of the indicators whose values are 0, 1. The Pearson correlation coefficient (\(\rho\)) is used to explain the relation between the drop-out label and each of the indicators whose values are continuous. The Pearson correlation coefficient is given by:
\[
\rho_{X, \text{dropout}} = \frac{\text{cov}(X, \text{dropout})}{\sigma_x \sigma_{\text{dropout}}}
\]

where \(X\) is a vector containing the values of the feature, \(\text{dropout}\) is a vector with the drop-out values \(\{0, 1\}\), \(\text{cov}\) is the covariance of each of the features and the label, and \(\sigma\) is the standard deviation.

**Production mode**

The testing mode is useful for checking the hypotheses of the EDM / LA researchers and to verify existing educational theories. Once a model is switched to production mode, actionable insights are generated from predictions. This production mode switch affects the whole site: Insights are now generated for all the users in the site. E.g. Arts teachers do nothing to receive predictions about their potential dropping-out students. They would receive notifications about these students once the prediction model is globally enabled for production mode.
As part of the Target definition, researchers can specify which predictions returned by the Machine Learning backend are worth observing and which predictions can be ignored. Actionable insights can be specified as part of the Target definition as well. These actionable insights are provided as suggested actions to the user that have access to predictions, usually a teacher in a course. Examples of actionable insights can be to message the at-risk student, to check the student activity log or to ignore the prediction. Figure 3.4 shows an example of how insights are presented to teachers.

The state of the Machine Learning algorithms is saved once the training process finishes. The trained algorithm is restored every time the Machine Learning backends need to generate predictions for ongoing courses.

With the production mode enabled, the system is continuously re-trained (using incremental learning techniques) when new courses are finished.

4.4 Case study

In this section, a prediction model generated using the Python Machine Learning backend included in the Supervised Learning framework is introduced. The presented model uses assessment to identify students at risk of dropping out of a MOOC. The indicators used in this model are based on the principles of the social constructivism and constructionism learning theories. They are all related to active activity generated by the user. In other words, activities and interactions where the student acquires knowledge by doing instead of by just reading. These variables were selected for the prediction model so as to study how assessments or the lack of them influence drop outs.

We implemented a Time splitting method to generate three predictions for the course duration. The first prediction is generated immediately when the first quarter of the course is over, using activity data from the start of the course. The second one is generated after half of the course is completed, using activity data from the beginning of the course up to that point. The third one is generated immediately after the third quarter of the course, also using all the data available from the start of the course up to that point in time. The selection of these three predictions is for functional purposes so teachers can get predictions regularly.

The actionable insights for teachers generated by this prediction model are shown in Figure 3.4. The actions executed by teachers are recorder for reporting purposes.

Dataset

To test the prediction model, we used the data from eight finished and anonymised MOOCs. They are 4-week long MOOCs offered by Moodle HQ twice a year. The course title is Learn Moodle and it is designed for anybody who wants to use the Moodle learning platform for teaching. Each of these eight MOOCs has between 4,027 and 6,233 students. Altogether they cover a total of 46,895 students. The dataset was anonymised by not only removing all trace of personal data but also randomizing the contents of all text fields, including the course name and its metadata.

The average drop-out ratio of these eight courses was found to be 82.55%, so the dataset is unbalanced. The F1 score rather than the accuracy would therefore be the best measure to describe the performance of the model [44].
The training dataset generated from the MOOCs contains a total of 115,353 rows which is approximately the number of users in the platform multiplied by 3, the number of predictions to be generated. Data cleansing techniques were applied to ensure the correctness of the data:

- Students whose enrolment dates do not match the course start and end dates were discarded.
- Students who never accessed the course were discarded.
- Activities whose due dates were not consistent with the course start and end dates were discarded.

As a binary classification problem, the samples where students have activity logs during the last quarter of the course are labelled as $0$, and samples which students do not have activity logs during the last quarter of the course are labelled as $1$.

**Indicators**

These are the Indicators that we used to identify students at risk of dropping out:

**Positive ratings in database entries**

The database activity in Moodle has strong fundamentals on the constructionist theory as other students in the course work together towards constructing a database of entries about a particular topic. Any student can add database entries and other students in the course can rate the entries positively or negatively, depending on the scale assigned to the database activity. Adding entries to a database means that the student has some degree of engagement to the course activities and that the student’s probability of dropping out of the course should be low. This variable measures how receiving positive assessments from other students influence in course drop-outs.

**Positive ratings in forum posts**

Forum posts are one type of activities in Moodle that have a strong learning theory base. Forums are organized in discussions and other students in the course can rate other students’ posts. Similar to the indicator above, regular postings to a forum are a sign of engagement in the course. In order to evaluate the influence of other students’ assessments in course drop-outs, the presence of positive ratings in forum posts that a student receives from other students is used as an indicator.

**The student attempted at least one quiz**

Moodle quizzes are automatically assessed and students get feedback pre-set by the teacher according to their answers and the grade. Attempting a quiz is a sign of engagement in the course.

**Percentage of quizzes attempted in the course**

Each section of the studied MOOCs contains a quiz activity. This variable calculates the frequency of attempts that the student made in pursuing assessment of their knowledge and skills.
The student failed a quiz attempt

The purpose of this indicator is to capture how negative assessments affect the student determination to complete the course.

Results

Results provided by the framework’s testing mode are shown in Table 4.1. Our test results show that the proposed model for students at risk gave an average accuracy of 88.81% and an average F1 score of 0.9337. As mentioned earlier, the studied dataset contains a 82.55% of drop outs. Therefore, the F1 score is a better representation of this model performance than the accuracy is.

The Python Machine learning backend included in Moodle LMS was selected for this study. We observed that the model performance did not change after 50 epochs so the number of epochs was set of 50 and the model was trained in batches of 1000 samples. The model was evaluated 10 times. The results highlighted above are the average of these 10 evaluations. We selected 10 as the number of repetitions by leveraging results stability and the time required to train the networks.

Test data includes samples that have been calculated using the three different time frames specified in our time splitting method. The results of our tests show that the accuracy of the predictions increases as more data is available. However, from a functional point of view, we think that a single measure of accuracy better represents the overall performance of the prediction model.

A sub-set of the indicators used in this study have a high and negative correlation to course drop outs. They explain the dependent variable in a great extent. These variables are: "Positive ratings in database entries”, "Positive ratings in forum posts”, "The student attempted at least one quiz” and "Percentage of quizzes attempted in the course”. The indicators "Positive ratings in database entries” and "Positive ratings in forum posts” are based on assessments received from other students. A possible interpretation for this correlation is that receiving positive assessments from other students encourages student to continue the course. Another possible interpretation is that students who pursue the completion of the course put more effort into their contributions to the course. The indicators "The student attempted at least one quiz” and "Percentage of quizzes attempted in the course” also have a strong correlation to course drop outs. A possible interpretation is that students who are interested in assessing their knowledge of the subject are also interested in completing the course.

The indicator "The student failed a quiz attempt” resulted to have less significance than expected. This variable, considered independently from the other variables used in this study, is not very relevant in these MOOCs when comparing it to the other variables. The purpose of this indicator is to capture how negative assessments affect the student determination to complete the course. Given these results we can not assure nor discard any possible relation. The contingency table of the variable and the drop-out label has also been studied without any interesting finding. Table 4.2 shows a summary of each variable correlation to drop-outs.
Table 4.1: Prediction model evaluation results

<table>
<thead>
<tr>
<th>Average accuracy</th>
<th>Average precision</th>
<th>Average recall</th>
<th>F1 score</th>
<th>AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>88.81%</td>
<td>91.51%</td>
<td>95.31%</td>
<td>0.9337</td>
<td>73.12%</td>
</tr>
</tbody>
</table>

Table 4.2: Indicators’ correlation to drop-outs.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Correlation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive ratings in database entries</td>
<td>MMC = −0.5567</td>
</tr>
<tr>
<td>Positive ratings in forum posts</td>
<td>MMC = −0.4472</td>
</tr>
<tr>
<td>The student attempted at least one quiz</td>
<td>MMC = −0.5563</td>
</tr>
<tr>
<td>Percentage of quizzes attempted in the course</td>
<td>ρ = −0.61994</td>
</tr>
<tr>
<td>The student failed a quiz attempt</td>
<td>MMC = −0.1101</td>
</tr>
</tbody>
</table>

4.5 Discussion

The presented framework aims to fulfil the need of advance analytics tools for Moodle. Another tool that aims to fulfil the same need is Moodle Data Mining (MDM), [56]. MDM is a complete tool that allows teachers to use data mining algorithms in their courses. It also provides a set of indicators and it supports classification, clustering and associations. However, it is a passive tool that relies on the teacher to generate the models. The presented framework is active and do not require any intervention from the teachers to start generating insights. The framework also deals with the usual caveats in analytics tools like scalability, and the cron jobs scheduling and partitioning. A limitation of the presented framework is that PHP coding is still required in some cases. The framework provides a set of indicators and time splitting methods. However, new predictive models need to implement a target class in PHP.

The presented model to identify students at risk of dropping out of a MOOC can be applied to Moodle courses that are offered periodically. The course activities should include activities such as forums, quizzes or databases as the indicators used in this study are based on such activities. The list of indicators used by the predictive model can be modified using the Moodle web interface. The framework provides alternative indicators that cover all the Moodle activities. The fact that all the past courses in Moodle are used to train the machine learning algorithms can be considered a limitation in some cases, as the content or activity generated can vary significantly among courses. The ability to create different predictive models for different sets of courses is part of the Moodle analytics roadmap for 2019. The courses should have both a start and end date. This limitation excludes open-ended courses from the training set.

4.6 Conclusions

The aim of this study is to present a framework for supervised learning that is integrated in the Moodle LMS. To create a prediction model using the presented framework is an advantage over creating a prediction model from scratch as the framework provides a set of base elements and a number of evaluation metrics to test the prediction models accuracy before enabling it for production. The
Supervised Learning framework presented in this paper is part of the Moodle LMS from version 3.4.0 onward, and it is available for free for the Moodle user community.

A case study model to identify students at risk of dropping out of a MOOC is described to illustrate uses of the presented framework. A total of 46,895 student enrolments in eight finished and anonymised MOOCs, offered as Moodle courses, have been analysed as part of the study. The accuracy and F1 score of the model are high and good for a production system despite using a very reduced set of indicators. According to the results obtained in this study, the activity generated by students on activities that provide automatic assessment contain enough information to identify students at risk of dropping out of a MOOC. The indicators used in this model are based on Moodle activities. The extra indicators required for the presented model have been bundled into a Moodle plugin which is publicly available in https://github.com/dmonllao/moodle-local_assessesindicators. The indicators have been included in Appendix A.

4.7 Future work

Future lines of research include reviewing the Learning Analytics and the Educational Data Mining literature for other suitable indicators that can be included in the framework. The presented model can be re-evaluated including new indicators.

Other machine learning and deep network techniques can also be investigated. Recurrent Neural Networks [55] can process the traces that students leave behind as time series which can potentially lead to improvements in accuracy. Other possible lines of research can explore the timing of the predictions generated by the system and how the prediction models accuracy changes as more data is included.

An ongoing line of work is focused on adding extra customisation capabilities to the predictive models using the Moodle web interface. Reports on the efficacy of each model will be added. These extra customization options include, but are not limited to:

- The ability to create different predictive models for different sets of courses.
- The name of the insights that are sent to users.
- The subset of users that receive the insights.
- The periodicity of the predictions, as well as the data included to calculate the indicators.

Another ongoing line of work is focused on reporting on the predictive models efficacy.

4.8 Acknowledgements

This research project was funded by Moodle Pty Ltd, and by the Australian government and The University of Western Australia through the Research Training Program (RTP). We thank Moodle HQ for providing the dataset used in this study. Special thanks for Helen Foster and Mary Cooch for setting up the MOOC and for running regular versions of the course. Also thanks to all Moodle HQ staff and members of the Moodle community that participated in the project by doing code reviews, by testing the framework and by helping design the user interface of the tool.
Chapter 5

A Quest for a one-size-fits-all Neural Network: Early Prediction of Students At Risk in Online Courses

5.1 Introduction

In this paper our focus is on studying a one-size-fits-all approach for early prediction of students at risk of missing assignments’ due dates in online courses.

Being able to get an early prediction of students who might not submit their assignments on time is important for many learning and education systems. It allows teachers and automated systems to intervene before it is too late. To miss an assignment due date or to send a late submission can have multiple consequences in the short term or in the long term, from getting a zero or a low grade in that specific assignment to losing overall interest to the course. The fact that a student misses an assignment submission has also been identified as an early indicator that the student is at risk [52, 64].

The term one-size-fits-all describes the ability to fit in all instances. The idea of finding a one-size-fits-all model that accurately predicts students at risk in online courses is controversial and there are studies raising concerns about how effective predictive models portability (also referred to as transferability) across different courses can be [16, 29]. Earlier publications in Learning Analytics and Educational Data Mining fields have studied the correlation between different variables that are based on students’ activity records and students’ course grades [40] or course abandonment [12, 38]. The variables that have a reasonably high correlation to students’ grades or course abandonment can be considered good predictor variables, understanding predictor as a variable with a high positive or negative correlation to the dependent variable. These variables are the most relevant variables we can use to predict students at risk. Many of the independent variables that are used for identifying students at risk highly depend on the online course context, e.g. the course structure or online activity assessment. Predicting students at risk in a course is not a trivial task as the predictions are dependent on the context:

- The values taken by independent variables may be in different ranges on different datasets, making the data difficult to normalized and also less adaptable to unseen data. The different
value ranges also make it difficult for comparison across different datasets.

- Different course workloads: When making predictions based on students’ activity records, the predictors need to take into account that not all the courses require the same amount of activity to be successfully completed. E.g. Difficult courses may require more students’ activity than easy courses.

- Fully-online courses versus blended courses: Fully-online courses have all students’ activities recorded online whereas blended courses include some face-to-face activities. The former type of courses usually record students’ grades online whereas the latter type of courses tend to have students’ grades recorded elsewhere. Predictor variables that rely on students’ grades could therefore not be applicable for blended courses.

A one-size-fits-all predictive model can hardly compete with a predictive model for a specific course, where predictions would be based on previous academic years data of that same course and independent variables would be specific to the course structure and content. However, online courses do not always have enough data to feed Machine Learning algorithms as previous versions of that same course may not be available or may not contain enough activity logs. A one-size-fits-all model is an interesting idea worth exploring as predictions could be generated for any online course in the LMS.

The focus of this paper is not on evaluating which of the activity-record-based variables available in the literature better correlate to missing assignments or late assignment submissions. Instead, we focus on developing a one-size-fits-all predictive model. The idea behind this study is to use neural networks to capture the hidden relations between input variables in order to contextualise predictor variables and to improve the predictive models’ portability. The predictive models presented in this paper only observe the students’ activity records up to two days before the assignment due date, the reason is that a production-ready system using this predictive model would be able to generate predictions two days before assignments due dates, giving them some time to take intervention. These interventions can take different forms, from notifying the teacher about the students that will likely not submit the assignment on time so him can individually discuss further issues with them, to sending a reminder directly to the student.

This paper is organised as follows: The datasets used to conduct this study are explained afterwards. Section 5.3 describes the input features used by our predictive models and Section 5.4 portrays the neural networks that get fed with these input features. The paper continues with the Evaluation used to evaluate the predictive models and the Results obtained. Finally, conclusions and some possible lines for future work are highlighted.

Contribution

Our research contributions are twofold and focused on the portability of predictive models for students at risk in online courses:

- Course and activity information has been included as input features and automatically-generated features from course-wise means, dataset-wise means and students’ activity predictor variables have also been included. This extends the set of variables used in similar studies of predictive
5.2. Datasets

<table>
<thead>
<tr>
<th>Kind of education</th>
<th>Total assignment submissions</th>
<th>Missing or late submissions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dataset 1</td>
<td>Online Training</td>
<td>1843</td>
</tr>
<tr>
<td>Dataset 2</td>
<td>School</td>
<td>3624</td>
</tr>
<tr>
<td>Dataset 3</td>
<td>High School</td>
<td>5052</td>
</tr>
<tr>
<td>Dataset 4</td>
<td>University</td>
<td>94763</td>
</tr>
<tr>
<td>Dataset 5</td>
<td>University</td>
<td>30323</td>
</tr>
</tbody>
</table>

Models portability [16, 29, 42]. To contextualise the variables that are based on students’ activity records results in better prediction accuracies according to the results of our tests. How adding context affects predictive models’ accuracy has been evaluated empirically.

- A set of neural network architectures, mixing predictor variables that are based on students’ activity records with context information have been compared. The predictive power of a subset of the presented Neural network architectures is higher than traditional fully-connected neural networks.

5.2 Datasets

To study the portability of predictive models in online courses, not only a large number of online courses are needed, but the courses need to be of wide variety. Therefore, the content of multiple Moodle sites used for different kinds of education have been analysed: University, school teaching and online training academies. The datasets do not belong to one specific area as they contain courses from different areas of study. Moodle Pty Ltd. and their partner institutions have provided these datasets.

Table 5.1 lists all the datasets used in this study. It is important to note that most datasets have imbalanced classes. The total samples from all the datasets result in a 64.66% of missing or late assignment submissions so classes can be considered balanced.

Labels

As a binary classification problem, two different labels are used: the samples where students have submission events in the activity log before the assignment due date are labelled as 0 and samples which students do not have any submission event in the activity log before the due date are labelled as 1. Past assignment activities from all the datasets listed in Table 5.1 have been analysed. When new assignment activities are created, an assignment submission record is added to the Moodle database for each student in the course. These assignment submission records include the submission status, with ‘empty’ being its default value. The assignment submission record is unique for each student and each assignment activity. Therefore, there is one sample for each activity and for each student in each dataset even if the student submits multiple versions to the same assignment activity (the assignment submission status is always the last submitted version).
Data cleansing

Data cleansing is a time-consuming process. Institutions adapt Moodle to their own needs and, therefore, use Moodle features in different ways. This wide variety of usages results in new challenges when calculating meaningful features from raw data. An example is: courses were reused across different academic years with previous academic years students enrolled in the course being kept while new student enrolments were added. This resulted in all students having the same permissions over the course contents, including the permission to add new assignment submissions. Another example is reusing courses where assignments’ due dates were not updated to the latest academic year of the courses.

The large number of studied datasets and the large number of courses in each of the datasets made it impossible to manually fix these datasets. Instead of manual fixing, and despite missing a portion of all the assignment submissions that could be studied, an automated process to discard courses, assignments and student enrolments that could lead to misleading results has been developed. Below are the criteria used to discard assignment activities and assignment submissions:

- Courses without a start date.
- Assignment activities without a due date.
- Assignment activities that are not visible to students. They have been discarded because students can not submit work.
- Assignment activities without coherent due dates. E.g. The due date is before the course’s start date.
- Assignment activities that require team submissions\(^1\). They have been discarded because not all team members have a submission activity log.
- Assignment submissions from students whose enrolment finished before the course start date. These submissions have been discarded to prevent old enrolments to interfere with the latest academic year’s data.
- Assignment submissions from students whose enrolments started more than one year after the assignment activity due date. These submissions have been discarded for the same reasons explained in the previous point. We have evaluated the risk of discarding these submissions as they are valid submissions for courses that last for more than one year. However, we concluded that discarding them is acceptable as the risk of corrupting the study results with old enrolments is higher than the benefit of considering these submissions as valid.

5.3 Input features

This section describes the independent variables used to predict missing or late assignment submissions. This study is not centred on evaluating the correlation between different independent variables and

\(^1\text{https://docs.moodle.org/35/en/Assignment_settings#Groups_submission_settings}\)
the label as mentioned in Section 5.1. Instead, we select variables on a subset of the most popular predictor variables available in the literature. Not all the independent variables have a good correlation to missing or late submissions. The term feature has been used as this is how the independent variables are usually referred to in Machine Learning.

Data sparsity affects predictive models’ performance negatively as training and testing samples are incomplete. Unfortunately, a subset of the input features described in this section are sparse so it is important that the trained neural networks can deal with sparse features. One option would be to discard all sparse features; this solution is not ideal as it would miss features with high correlation to the label. Data normalization would be another option although it would lead to negative collateral effects given the high percentage of empty values. The adopted solution to deal with data sparsity is to limit all input feature values to fit into a $[-1, 1]$ range, leaving zero values for empty features.

The input features described in this section are calculated using a Moodle plugin\textsuperscript{2} developed as part of this study. The plugin was made publicly available in the following code repository: https://github.com/dmonllao/moodle-local_latesubmissions.

**Students’ activity (SA)**

Students’ activity records are the main source of information in the LMS. The features listed in this section correlate positively or negatively to the label. These predictor variables that are based on students’ activity records have been partly extracted from the Learning Analytics and Educational Data Mining literature [16, 66] and reformulated while others have been added to balance particular circumstances described in this section.

These features are calculated using the activity logs from the start of the course up to two days before the assignment due date. This is very important as procrastination is a factor to consider and it affects some of the input features listed in this section. Different time frames were evaluated before we settled on using two days prior to the assignment’s due date as the time frame used in this study. To report about how the predictive power decreases as we go further from the due date would require a separate paper as it is a complex topic that requires a specific literature review. As a summary, we found two days prior to the assignment’s due date to be the time frame that best balances the impact of procrastination over the predictive power of the model with enough time for the teacher or the student to intervene.

**Assignment activity accesses by the student**

The rationale behind this predictor is that it is not likely that students are going to submit assignments if they have not even accessed them. Instead of using an absolute number of accesses, the number of accesses has been categorized into no accesses, less than three accesses per week and more than three accesses per week. These categories are numerically expressed as $\{-1, 0, 1\}$.

\textsuperscript{2}https://docs.moodle.org/dev/Plugin_types
The student has not previously accessed the course

This input feature is quite self-explanatory. Students who never accessed the course are not likely to access it now. The feature values are \([-1, 1]\).

Accessed activities and resources in the same course section

Moodle courses are organized in sections\(^3\). They are used to group activities and resources that are related to each other. Another possible use of course sections is to split the course by the course duration and to have a section for each week of the course. The more related activities and resources students access, the more likely it is that they are going to actively participate in the assignment activity. The value of this feature is converted to the \([-1, 1]\) range using the following formula: \((2 \cdot \text{accesses}) - 1\), where \(\text{accesses} = 1\) if all activities have been accessed, \(\text{accesses} = 0\) if no activities have been accessed and a value in the \([0, 1]\) range representing the percentage of activities in that section that were accessed by the student.

Level of passive activity generated by the student in the course

The number of course accesses is one of the predictors that the Learning Analytics literature found to correlate well to students at risk. The word \textit{passive} in the section heading means that only \textit{view} activity logs are observed. Instead of using an absolute number, the activity has been categorized into \textit{no activity}, \textit{less than three activity logs per week} and \textit{more than three activity logs per week}. These categories are numerically expressed as \([-1, 0, 1]\).

Level of active activity generated by the student in the course

Similar to the \textit{level of passive activity generated by the student in the course} feature, this feature captures how intensive the student activity has been until two days before the assignment due date. This feature focuses on the \textit{write} activity logs. So it captures forum posts, student comments and other activities posts and submissions. Instead of using an absolute number, the activity has been categorized into \textit{no activity}, \textit{less than three activity logs per week} and \textit{more than three activity logs per week}. These categories are numerically expressed as \([-1, 0, 1]\).

The user profile is complete

The user profile page in Moodle allows users to set up their profile picture, a description, a list of interests and links to their favourite social networks. This predictor variable reflects the level of completeness of the student user profile. A complete user profile results in high values and a default and empty user profile results in low values. Circumstances like student suspensions or missing policy agreements are also considered and result in low values as well. The user profile attributes that are observed are mainly the user picture, the description and the user interests. The feature values are in the \([-1, 1]\) range.

\(^3\text{https://docs.moodle.org/35/en/Course_homepage}\)
The student attempted quizzes close to their due date

Procrastination is one of the biggest challenges early prediction systems face. This feature was added to counteract the effects of students’ procrastination. The Moodle quiz module allows teachers to build quizzes, optionally setting a due date, if quizzes do have a due date the proximity of the submission to the due date the student attempted the quiz can be observed. This provides some insight about the student’s tendency to procrastinate. The feature is calculated by categorizing the attempts to each quiz into five groups: No attempts, attempts after the due date, attempts the day before the quiz due date, attempts two days before the quiz due date and attempts more than two days before the quiz due date. The value is calculated by averaging all quizzes in the course.

The student submitted choice activity answers close to their due date

Moodle has a choice activity that allows teachers to add a list of options and students must select one of the options. Choice activities may not add much educational value to a course, but they also have a due date and so they can be used, similarly to how quiz activities have been used in The student attempted quizzes close to their due date feature, to infer the student’s tendency to procrastinate. The feature is calculated like the feature mentioned in Section 5.3.

The student submitted other assignment activities close to their due dates

This is the student attempted quizzes close to their due date feature for assignment activities. The assignment activity that is the target of our predictions is skipped from the list of observed assignment activities. The feature is calculated the feature mentioned in Section 5.3.

Other assignment activities the student sent a submission to

There may still be misclassified samples due to students’ procrastination despite our efforts to counteract it. This input feature is related to the student submitted other assignment activities close to their due dates feature as it observes other course assignments. In this case the percentage of assignment submissions by the student to other assignment activities in the same course is calculated. The feature is calculate the feature mentioned in Section 5.3, replacing accesses by assignment submissions.

Course and assignment information (CAI)

Students’ activity features may not be enough to make predictive models portable to unseen datasets. The assignment activity and the course it belongs to are subject to a set of circumstances that make predictive models portability a complex issue:

- Some courses are fully online and some have a face-to-face part.
- Course contents and workload differences: Using variables based on students’ grades has been avoided. Even if a course is not difficult it may require a lot of students’ activity to be successfully completed while others may require less activity.
Courses that are part of a degree or certification versus free Massive Open Online Courses (MOOC): Abandonment rates differ between High School online courses and free MOOCs [2, 3].

An ideal one-size-fits-all neural network, given the appropriate input features, should be able to internally abstract the key elements that make the same set of values for predictors based on students’ activity records return different labels when the context is different. This is the purpose of the following list of course and assignment information features.

The weight of the assignment in the course

The weight of an activity in the course can be a good measure of how important the activity is in the course. The more important an activity is in a course, the more a student may be willing to send a submission. Moodle does not allow teachers to specify when an activity is required or not, so activities with no weight can be considered optional and students may be more inclined to skip them than to skip the required activities. The activity weight is categorised into the following scales: no weight, less than 10% of the course weight, less than 20%, less than 50% and more than the 50%.

Completion tracking is enabled for the assignment activity

Moodle features an activity completion tracking system that allows teachers to specify when an activity can be marked as completed. This feature is usually used alongside other features like access restriction or course completion so an activity with completion tracking enabled is likely to be a requirement for another activity or a requirement to complete the course. The feature values are \{-1, 1\}.

Students need to obtain a minimum grade for the assignment activity

Moodle allows activities to have a grade-to-pass value. This is the minimum grade a student must achieve so the activity is considered passed. As with the weight of the assignment in the course and completion tracking is enabled for the assignment features, this feature is a measure of how required an activity is. The feature values are \{-1, 1\}.

The course teacher set a grading due date for the assignment activity

Teachers can set a grading due date. Assignment activities that are required to complete the course need to be graded by teachers. Setting a grading due date can be good indicator that the teacher is going to grade student submissions. The feature values are \{-1, 1\}.

Guest access is enabled in the course

A course with guest access enabled allows any user to access the course contents. The feature values are \{-1, 1\}.

Self-enrolment is enabled in the course

A course with self-enrolment enabled allows any user in the Moodle site to enrol in the course. The rationale behind adding this feature is that courses that are part of a degree, a curriculum or a set
of courses do not usually allow users to be self-enrolled. Students may be more inclined to complete activities in courses that are part of an official degree or curriculum. The feature values are \{-1, 1\}.

The student self-enrolled into the course

Related to *Self-enrolment is enabled in the course*. feature and with its rationale. A self-enrolled student that is not part of a degree or curriculum may feel less bounded to the course and its completion. The feature values are \{-1, 1\}.

Number of submission attempts

Assignment activities can limit the number of submission attempts each student can send. Assignments with no limits to the number of attempts may be optional or due dates may be more flexible. Assignments that accept a low number of attempts may be more restrictive in other aspects as well. The setting value is transformed to the following scales: *No limits*, *one attempt*, *two attempts*, *more than two attempts*.

The assignment activity has a cut off date

Assignment activities in Moodle can have a due date and a cut-off date. If both dates are set submissions are allowed until the cut-off date. Students may defer their assignment submissions to the cut off date if the assignment activity has both a due date and a cut off date. The feature values are \{-1, 1\}.

Students accepted a submission statement

The assignment activity can be configured so that students need to accept a submission statement before sending a submission. This feature may be irrelevant although it could also indicate the formality required by the assignment activity. This statement is usually used to stress the plagiarism policies. The feature values are \{-1, 1\}.

Course teachers are notified about new student submissions

Moodle allows teachers to be notified when there are new student submissions. Teachers may not be interested in receiving notifications about assignment activities that are not important for the course. The feature values are \{-1, 1\}.

Students are notified about teachers’ feedback

Moodle allows students to be notified when there is new feedback from the teacher. Teachers may disable this assignment activity setting if they do not plan to provide feedback to students. The feature values are \{-1, 1\}.

Peers activity (PA)

We tried to avoid making assumptions about the course when formulating the SA features described in Section 5.3. Despite our efforts there may still be some hidden assumptions like the fact that courses
require regular accesses. The activity logs generated by other students in the course are valuable information and they provide information about the context where SA features are calculated. This activity also helps the neural networks balance SA features taking into account the course workload [42].

Our algorithm automatically adds four extra features to the input dataset for each SA feature. These extra features are:

- The mean of all the students in the course. E.g. Paul accessed a course twice, Maria accessed the same course 6 times and Nicole 4 times; the mean is 4 so an extra feature to all 3 samples with value 4 is automatically added. This extra feature purpose is to add information about the course peers.

- The mean of all the students in the dataset. E.g. Paul accessed 'Course 1' 2 times, Maria accessed 'Course 2' 10 times and Nicole accessed 'Course 3' 6 times; the mean of all students in the dataset is 6 so an extra feature to all 3 samples with value 6 is automatically added. This extra feature purpose is to add information about the institution as different institutions use the LMS in different ways.

- The SA feature value relative to the course peers. This feature value is defined as $(SA_{ij} - CP_{i}) \div 2$, where $i$ is the SA feature index, $j$ each sample value for SA$_i$ feature and CP$_i$ is the course mean for SA$_i$ feature.

- The SA feature value relative to all the students in the dataset. This feature value is defined as $(SA_{ij} - DP_{i}) \div 2$, where $i$ is the SA feature index, $j$ each sample value for SA$_i$ feature and DP$_i$ is the dataset mean for the SA$_i$ feature.

### 5.4 Algorithms

Neural networks [78] build relations between input features while they are trained so they are able to predict the correct label. They improve their accuracy during training by back propagating [79] the difference of the calculated label of a set of sample versus the real values to the network’s previous layers, updating its weights. The connections between different layers become stronger or weaker after the back-propagation process.

**Overfitting**

The input features are not always as well-correlated to the label and neural networks often have a hard time fitting the input features to the label. Features can be very sparse, there can be irrelevant features or features in specific datasets can be tied to specific values. These input features are a challenge for a neural network and the neural network struggles to update its weights according to these situations but it is still able to do it. The problem is that this can easily lead to weights overfitted to the training datasets and therefore to algorithms that are not good at classifying samples from an unseen dataset. The more sparse and irrelevant input features in the studied datasets are, the more likely that the algorithm would overfit the training data.
The context features described in Section 5.3 can be relevant for some testing datasets and irrelevant for other testing datasets. Features added to infer how compulsory the assignment activity is can be irrelevant in University courses that are part of a degree if the institution offering the courses does not bother to enable completion tracking or if they do not manage grading through the University LMS. These same features are relevant when the institution offering the course or the course teacher do set these values. Other features can be very sparse as they depend on specific activity types like the choice activity or the quiz activity that are only present in some courses. We are evaluating how effective a one-size-fits-all neural network is, so overfitting is a critical issue that needs to be overcame.

Overfitting is a common issue in classification [81] and it is not unique to our classification problem.

**Dropout regularization**

This dropout regularization method [81] discards random connections between units in different layers so that networks are forced to strengthen or weaken connections between the rest of the weights. We applied a 20% dropout to all the hidden layers weights.

**Weights regularization**

To regularize our weights [65] closer to zero reduces overfitting as having large weights make the networks too sensitive (small changes in the input features result in significant changes in the output). The studied datasets contain a high number of sparse features. Tanh (the activation function used in this study) is zero centred and the empty values in the studied datasets are represented by zero. We want the empty values to be propagated to further layers in the network. Therefore, L1 regularization is used to regularize the network weights as it does not alter zero values like L2 regularization does. E.g. The feature in Section 5.3 is null (zero value) if the course does not contain choice activities. Weight regularization has been applied to context-related layers.

**Studied neural networks**

Previous studies available in the Learning Analytics and Educational Data Mining literature only use predictors based on students' activity records, a neural network with predictor variables based on students' activity records has been used as the baseline. The prediction results from networks that include context features (CAI features and PA features) and other network architectures described in this section are compared to this baseline network to evaluate how effective our proposed improvements are.

All the networks are set up as follows:

- One unit in the input layer for each input feature.
- One single hidden layer with 20 units is used unless otherwise stated.
- One single unit is used in the output layer.
- tanh is used as activation function unless otherwise stated.
- Adam optimization [47] is used in the training iterations.
Weights are initialized using Xavier initialization [32].

- **Batch gradient descent** (using the whole dataset for each weights update) is used.

- A dropout regularization of 20% is applied to each fully-connected layer.

### Students’ activity only

As mentioned in the previous section, a basic neural network with predictors based on students’ activity records as input data will be used as our baseline. This basic neural network architecture is the traditional feed-forward neural network with fully-connected layers and one single hidden layer described in Section 5.4. This baseline network defines the portability of this predictive model for missing or late assignment submissions using the traditional approach where only \( SA \) features are used as input data.

### Student and peers activity

Adding \( PA \) features to each \( SA \) feature is a way to add context without adding \( CAI \) features that could bias results or make algorithms overfitted to the training data. This network is also feed-forward and units between layers are fully-connected as in network 1: Students’ activity only although in this case a second hidden layer, also with 20 units, has been included so the network can model more complex relations between variables.

### All input features

This network includes all features described in Input features section: \( SA \) features, \( CAI \) features and \( PA \) features. This approach assumes that the neural network is able to build the relations between input features by itself. The number of possibly irrelevant \( CAI \) features is one of the possible problems this network can have. This network is also feed-forwarded and units between layers are fully-connected as in network 1: Students’ activity only. A second hidden layer, also with 20 units, has been added so the network can model more complex relations between variables.

### Students’ activity contextualised through training

The idea behind this neural network is that we do not know how much the context affects \( SA \) features so we let the neural network work it out for us by adding an extra layer where each \( SA \) input feature...
is trained together with all context features. The number of units in this extra layer is equal to the number of $SA$ features. Although this architecture looks conceptually beautiful it implies a couple of undesired conditions: The number of weights that are related to context features is higher than the number of weights related to students’ activity records, this is not ideal as $SA$ features are the real missing or late assignment submission predictors. A related side-effect is that irrelevant $CAI$ features can significantly affect the model accuracy. A dropout of 20% is applied to the contextualised layer.

Figure 5.1 shows the extra set of weights that connects $SA$ features with context features (contextualized hidden layer). This extra set of weights uses the same activation function than the fully-connected hidden layer weights and the same dropout regularization percentage. The number of units in each layer in Figure 5.1 can be ignored, the purpose of this figure is to describe the network architecture. The real number of units in the contextualized hidden layer is equal to the number of $SA$ input features and the number of unit in the fully-connected hidden layer is 20.

Students’ activity contextualised through training alongside the original $SA$ inputs

This neural network is similar to Students’ activity contextualised through training, the difference is that this network tries to balance the possibility that context features can be irrelevant. Figure 5.2 shows how $SA$ features are fed together with the contextualised $SA$ features to the fully-connected hidden layer. On the one hand are the weights that connect $SA$ features with the fully-connected hidden layer units, these weights are trained exactly like they are trained in a fully-connected neural network with one single hidden layer. On the other hand are the same inputs contextualised through back-propagation. As in Students’ activity contextualised through training, the extra set of weights use the same activation function than the fully-connected hidden layer weights and the same 20% dropout regularization is applied to the contextualised layer. The architecture of this network is shown in Figure 5.2. The number of units shown in the figure are for illustration purposes only. The real number of units in the contextualized hidden layer is equal to the number of $SA$ input features and the number of unit in the fully-connected hidden layer is 20.
CHAPTER 5. A QUEST FOR A ONE-SIZE-FITS-ALL NEURAL NETWORK: EARLY PREDICTION OF STUDENTS AT RISK IN ONLINE COURSES

Figure 5.3: Students’ activity and context trained separately.

Figure 5.4: Students’ activity and context trained separately with one context hidden unit.

Figure 5.5: Students’ activity and context trained separately, considering peers activity as part of the students’ activity.

Students’ activity and context trained separately

To separate input features into 2 hidden layers that are later merged and connected to the main fully-connected hidden layer is another studied approach. This allows the network to model students’ activity and context separately. Three variants of this network have been studied:

- Network 6.1: Figure 5.3 shows the extra sets of weights (one for SA features and another one for CAI and PA together) connected to the main fully-connected hidden layer. The two extra set of weights use the same activation function than the main fully-connected hidden layer weights and the same dropout regularization percentage.

- Network 6.2: Figure 5.4 shows the same extra sets of weights than in network 6.1 but in this case the trained context layer have only one unit so the weight of context-related units is lower.
5.5 Evaluation

In this section the methodology for evaluating the performance of the neural networks detailed in Section 5.4 using the five datasets detailed in Section 5.2 is described.

Portability between courses in the same institution may not be a complete measure of a predictive model portability. Courses in the same institution tend to have the same course structure or share the same level of obligatoriness. Therefore, the portability of each of the studied neural networks has been evaluated both across different datasets and across different courses in the same dataset.

The measures used to compare the studied neural networks are *Accuracy* and *F1 score*. The accuracy of a predictive model is the fraction of predictions that the Machine Learning algorithm identified with the correct label. It is calculated as follows:

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}}$$
The accuracy value can be misleading if our datasets classes are unbalanced like they are in some of the datasets used in this study. The \textit{F1 score} is an alternative accuracy measure, with values in the \([0,1]\) range, that is better suited when the studied datasets contain unbalanced classes. It is the harmonic mean of \textit{Precision} and \textit{Recall}. \textit{Precision} is the fraction of the elements identified as positives by the Machine Learning algorithm that are really positives (true positives). \textit{Recall} is the fraction of the true positive elements the Machine Learning algorithm identified. If either of \textit{Precision} or \textit{Recall} is low then so will the \textit{F1 score}. The \textit{F1 score} will only be good if both \textit{Precision} and \textit{Recall} are good.

The evaluation process can be outlined as follows:

1. Generate one data file for each dataset (see Section 5.2).

2. Iterate through the five datasets (Table 5.1) and the seven networks described in Section 5.4 for the following steps:
   
   a) Evaluate the network using other datasets as training data:
      
      i. Train the neural network with all datasets but the studied one.
      
      ii. Test the trained network on the studied dataset and calculate the \textit{accuracy} and \textit{F1 score}.
      
      iii. Repeat 2.a.i) and 2.a.ii) steps and calculate the \textit{accuracy} and \textit{F1 score} mean.
   
   b) Evaluate the network using the studied dataset:
      
      i. Split the studied dataset into 90\% for training and 10\% for testing.
      
      ii. Train the neural network using the training set.
      
      iii. Test the network using the test set and calculate the \textit{accuracy} and \textit{F1 score}.
      
      iv. Repeat 2.b.i), 2.b.ii) and 2.b.iii) steps and calculate the \textit{accuracy} and \textit{F1 score} mean.

Steps 2.a.i) and 2.b.i) were initially run using different numbers of epochs, it was observed that the \textit{accuracy} and the \textit{F1 score} of the evaluated neural networks were stable around 1000 epochs. Therefore, 1000 was the final number of epoch used to train the neural networks. Steps 2.a.iii) and 2.b.iv) where run using different numbers of repetitions in order to reduce the variability of the results. 5 resulted to be the number of repetitions that better balanced execution time and constant results and therefore the final number of repetitions what was used in the evaluation process.

The process described in this section along with the neural networks developed for this study were implemented using Keras and Tensorflow. The Keras context models used in our experiments are publicly available in the following code repository: https://github.com/dmonllao/keras-context-models.
Table 5.2: Using unseen datasets as testing data (one-size-fits-all). Average accuracy and F1 score after running 1000 epochs and 5 repetitions.

<table>
<thead>
<tr>
<th>Dataset 1</th>
<th>Dataset 2</th>
<th>Dataset 3</th>
<th>Dataset 4</th>
<th>Dataset 5</th>
<th>Averages</th>
</tr>
</thead>
<tbody>
<tr>
<td>F1 Acc.</td>
<td>F1 Acc.</td>
<td>F1 Acc.</td>
<td>F1 Acc.</td>
<td>F1 Acc.</td>
<td>F1 Acc.</td>
</tr>
<tr>
<td>Network 1</td>
<td>0.663</td>
<td>0.897</td>
<td>0.613</td>
<td>0.866</td>
<td>0.678</td>
</tr>
<tr>
<td>Network 2</td>
<td>0.667</td>
<td>0.900</td>
<td>0.611</td>
<td>0.888</td>
<td>0.760</td>
</tr>
<tr>
<td>Network 3</td>
<td>0.542</td>
<td>0.894</td>
<td>0.597</td>
<td>0.887</td>
<td>0.740</td>
</tr>
<tr>
<td>Network 4</td>
<td>0.658</td>
<td>0.863</td>
<td>0.516</td>
<td>0.866</td>
<td>0.713</td>
</tr>
<tr>
<td>Network 5</td>
<td>0.616</td>
<td>0.753</td>
<td>0.612</td>
<td>0.887</td>
<td>0.725</td>
</tr>
<tr>
<td>Network 6.1</td>
<td>0.631</td>
<td>0.895</td>
<td>0.608</td>
<td>0.889</td>
<td>0.700</td>
</tr>
<tr>
<td>Network 6.2</td>
<td>0.550</td>
<td>0.891</td>
<td>0.596</td>
<td>0.891</td>
<td>0.742</td>
</tr>
<tr>
<td>Network 6.3</td>
<td>0.766</td>
<td>0.887</td>
<td>0.597</td>
<td>0.887</td>
<td>0.782</td>
</tr>
<tr>
<td>Network 7</td>
<td>0.747</td>
<td>0.989</td>
<td>0.610</td>
<td>0.887</td>
<td>0.785</td>
</tr>
</tbody>
</table>

Table 5.3: Splitting each dataset in training and testing. Average accuracy and F1 score after running 1000 epochs and 5 repetitions.

<table>
<thead>
<tr>
<th>Dataset 1</th>
<th>Dataset 2</th>
<th>Dataset 3</th>
<th>Dataset 4</th>
<th>Dataset 5</th>
<th>Averages</th>
</tr>
</thead>
<tbody>
<tr>
<td>F1 Acc.</td>
<td>F1 Acc.</td>
<td>F1 Acc.</td>
<td>F1 Acc.</td>
<td>F1 Acc.</td>
<td>F1 Acc.</td>
</tr>
<tr>
<td>Network 1</td>
<td>0.721</td>
<td>0.984</td>
<td>0.491</td>
<td>0.788</td>
<td>0.864</td>
</tr>
<tr>
<td>Network 2</td>
<td>0.848</td>
<td>0.980</td>
<td>0.675</td>
<td>0.786</td>
<td>0.864</td>
</tr>
<tr>
<td>Network 3</td>
<td>0.722</td>
<td>0.981</td>
<td>0.663</td>
<td>0.771</td>
<td>0.835</td>
</tr>
<tr>
<td>Network 4</td>
<td>0.655</td>
<td>0.983</td>
<td>0.651</td>
<td>0.755</td>
<td>0.823</td>
</tr>
<tr>
<td>Network 5</td>
<td>0.680</td>
<td>0.985</td>
<td>0.658</td>
<td>0.759</td>
<td>0.830</td>
</tr>
<tr>
<td>Network 6.1</td>
<td>0.744</td>
<td>0.988</td>
<td>0.524</td>
<td>0.784</td>
<td>0.863</td>
</tr>
<tr>
<td>Network 6.2</td>
<td>0.711</td>
<td>0.988</td>
<td>0.656</td>
<td>0.778</td>
<td>0.845</td>
</tr>
<tr>
<td>Network 6.3</td>
<td>0.836</td>
<td>0.983</td>
<td>0.718</td>
<td>0.749</td>
<td>0.845</td>
</tr>
<tr>
<td>Network 7</td>
<td>0.827</td>
<td>0.984</td>
<td>0.696</td>
<td>0.767</td>
<td>0.832</td>
</tr>
</tbody>
</table>

5.6 Results

Table 5.2 and 5.3 show the results obtained from the evaluation step 2.a) and 2.b) described in the previous section. The three largest values in each column are in boldface. The two networks that take up the 4th and 5th places on the Averages column are underlined.

According to our tests, network 2 (Student and peers activity), network 6.3 (Students’ activity and peers separated from course info) and network 7 (Students’ activity, peers activity and course information trained separately) are the networks with more predictive power. The baseline network 1 (Students’ activity only) predictive power is the lowest one (along with network 4 (Students’ activity contextualised through training)) in Table 5.3. The predictive power does not increase by just adding more and more CAI features to the predictor variables as we can see when comparing network 3 (All input features)’s results with other networks that balance the CAI features importance.

The reported F1 score and accuracy average results are relatively high in both Table 5.2 and Table 5.3. The differences between Table 5.2 and Table 5.3’s results in the Averages columns are between 0.1% and 5.9% for the F1 score and between 2.0% and 7.6% for the accuracy. Despite these differences are noticeable they do not represent a high percentage.

However, the results between datasets vary significantly:
In Table 5.2, the results on dataset 1 are below the results in the Averages column, and only the accuracies of network 6.3 and network 7 are above the 70%. In dataset 1 case, the difference between Table 5.2 and Table 5.3’s results are higher than the difference between Table 5.2 and Table 5.3 results in the Averages column. The network architectures where the PA features have an important weight result in higher predictive power.

In Table 5.2 and Table 5.3, the results on dataset 2 are above the results in the Averages column. Dataset 2 results remain similar regardless of the neural network architecture. In dataset 2 case, the difference between Table 5.2 and Table 5.3’s results is around a 10% which is higher than the difference between Table 5.2 and Table 5.3’s results in the Averages column. The courses in this dataset are fully online so all learning activities are tracked in the LMS. This could partly explain why the results on this dataset are higher than the results in the Averages column.

In Table 5.2 and Table 5.3, the results on dataset 3 have low predictive power compared to other datasets. A possible explanation is that the features used in this study have not been able to capture the aspects that makes students submit assignments on time. Another possible explanation is that the Moodle site this dataset is based on makes a blended use of the assignment module, accepting both online and offline assignment submissions. Offline submissions are not recorded in the LMS activity log, and thus offline submissions would be incorrectly labelled as late or missing assignment submissions, affecting the predictive model performance.

The results on dataset 4 in Table 5.2 are above the results in the Averages column. They remain similar regardless of the neural network architecture. It is the only dataset whose Table 5.3 results are lower than the Table 5.2 ones.

The results on dataset 5 in Table 5.2 are similar in all networks but in network 4 and network 5. These two networks are the ones that rely more on the CAI features.

Dataset 1 and dataset 3 should be studied in detail. Their study could result in new features added to the feature set or in discarding the datasets as some circumstances could invalidate the successful application of predictive models.

A subset of the studied CAI features are not direct predictors of the studied label nor had a clear relation to other predictor variables. They are irrelevant and they do not improve predictive power nor reduce it. The irrelevant features are: Students accepted a submission statement, The assignment activity has a cut off date, The course teacher set a grading due date for the assignment, Number of submission attempts accepted, Course teachers are notified about new student submissions and Students are notified about teachers’ feedback.

5.7 Conclusions and future work

The results presented in Section 5.6 show that adding context to predictors based on students’ activity records and using specific neural network architectures improve the network’s accuracy. Despite the relatively good accuracy of the presented neural networks, we can not claim that they fit all courses. The efficiency of a predictive model will always depend on how institutions and teachers use the LMS.
Further study is required before our one-size-fits-all model can be applied to a brand new Moodle site with the guarantee that its accuracy is good enough for a system in production. One of the weak points of the indicators used in this article is that some of them rely on hard-coded numbers that may prevent them to generalize well. A study limited to institutions that offer fully online courses could help clarify how far we are from the one-size-fits-all model.

Previous portability studies revealed different levels of variability. OAAI [42] reported accuracies in the 61.95-85.63% range, which is about a 20-25% of variability, these results are similar to this study results and the variability is similar as well. Other studies like [16] reported a higher variability between different courses. To limit our input feature values to a range of values and to apply regularization techniques helped neural networks generalize well to unseen datasets. The average performance decrease between networks trained with other datasets and networks trained with data from the same dataset is between 2.0% and 7.6% which represents a low percentage.

The baseline network 1 performance has been surpassed by other networks that include context (PA or CAI) features. PA features resulted to be more relevant than CAI features. Giving relative importance to the context and finding a good balance between SA features, PA features and CAI features is what resulted more effective according to our test results. Neural networks that put more weight over the context instead of over the students’ activity or the peers’ activity result in worst predictive power than just using SA features. Having more CAI features than SA features leads to overfitted models. The same happened with hidden layers which units are based on CAI features.

To expand the set of features to all (or most relevant) predictor variables available in the LA and EDM literature is one of the possible directions for future work. To add context variables like whether the course is fully or partially online could improve the predictive models performance as well. A related research direction to this could be focused on improving how the algorithms manage procrastination. Conclusions from relevant studies that cover this topic could be used as a foundation for future work in this area [10, 11].

Future studies could expand the predictive models evaluation process to train the neural network using all datasets but the one used for testing the algorithm like it was described in Section 5.5 in this paper but running an extra training process using a part of the tested dataset. Variations of this dual training process could include transfer learning [67].

Further studies including different or extra datasets could lead to improvements to the data cleansing processes. Another research line could be centred on the most effective interventions for students that will likely not submit assignments on time.

Acknowledgment

The authors would like to thank Moodle Pty. Ltd. partner institutions for providing anonymised datasets for this research. We would like to extend the thanks to the Moodle Pty. Ltd. employees that helped on data acquisition and data processing. We are grateful to the Nvidia Corporation for the support of a Titan Xp GPU, which we used to train and test our algorithms in this paper.
Chapter 6

Conclusions

Software, predictive models and new neural network architectures have been developed as part of this thesis. A software framework for the development and evaluation of supervised learning models is presented in Chapter 3. The framework is part of the Moodle LMS from version 3.4.0 onward, and it is available for free for the Moodle user community. The paper that formulates Chapter 4 is an extension of Chapter 3. A predictive model to identify students at risk of dropping out of a MOOC illustrates uses of the framework presented in the previous chapter. An extension for the Moodle LMS has been developed as part of this paper and has been released publicly in https://github.com/dmonllao/moodle-local_assessesindicators, under GNU General Public License v3.0. The input features of the model are based on automatically assessed activities in Moodle. In Chapter 5, the portability of predictive models is studied. Different neural network architectures are trained with multiple input datasets and tested with separate datasets, which were generated from an unseen Moodle site. Another extension for the Moodle LMS has been developed for this paper and has also been released publicly in https://github.com/dmonllao/moodle-local_latesubmissions using the GNU General Public License v3.0.

The presented framework provides a base for any LA and EDM researcher to create and evaluate new and portable predictive models in Moodle. The framework includes a set of reusable elements and a number of evaluation metrics to test the accuracy of prediction models. This is a significant improvement from creating predictive models from scratch as it speeds up the process. The framework allows site administrators and researchers to use their Moodle sites’ databases as a benchmark dataset by comparing the accuracy of the different predictive models generated by the institution researchers or imported from other institutions.

Once a model is enabled for production mode, the framework automatically generates insights so educators can easily identify and support students with difficulties. Therefore, this study is of great significance for all students using Moodle worldwide. The fact that the framework manages the whole life cycle of the prediction model, including the generation of insights for users, and the fact that the framework includes the automatic generation of features for portability purposes is an improvement over approaches like [56].

The framework has its limitations. New targets and indicators for the framework need to be coded in PHP, which can be a barrier for institutions interested in extending the predictive models included in Moodle LMS. Another limitation for some institutions is the fact that the indicators used to generate
predictions read activity data from the activity logs. Activity log tables are generally large and it can take time to read them. A relevant limitation for researchers interested in estimating linear variables is that regression analysis is not supported yet although there are short-term plans to add support for it.

A total of 125,617 student enrolments have been analysed and several predictive models have been developed as part of this thesis. The main findings are related to the portability of predictive models. A model to identify students who will likely miss an assignment due date has been developed for the study in Chapter 5. Different neural network architectures and different strategies to add context to the input variables have been compared. The results presented in Section 5.6 show that adding context to predictors based on students’ activity records and using specific neural network architectures improve the network’s accuracy. However, a study limited to institutions that offer fully online courses could help clarify how far we are from the one-size-fits-all model. These results support the results obtained in other mid-size studies like OAAI [42], where accuracies in the 61.95-85.63\% range were reported, which is a 20-25\% of variability. Studies like [16] reported a higher variability between different courses while other studies [29] directly advise against promoting one-size-fits-all approaches to the prediction of students at risk.

6.1 Future work

There are several future research directions for this project. The framework could be extended to cover unsupervised learning using the same 2-layer architecture. The framework has been designed to keep Unsupervised Learning in mind so the same set of indicators for Supervised Learning could be used for Unsupervised Learning. Another option for future study is to explore how cross-validation can improve the adaptability of the Machine Learning algorithms included in the Moodle LMS.

An ongoing line of work is focused on adding extra customisation capabilities to the predictive models using the Moodle web interface. Reports on the effectiveness of each model will be added. These extra customization options include, but are not limited to:

- The ability to create different predictive models for different sets of courses.
- The name of the insights that are sent to users.
- The subset of users that receive the insights.
- The periodicity of the predictions, as well as the data included to calculate the indicators.

Learning Analytics literature can be reviewed and extra student engagement indicators can be extracted from the literature. The effect of procrastination or the tendency to procrastinate of a subset of the students is also an interesting line of research due to the fact that this study shows a strong emphasis on solutions to real-world problems and that early prediction systems can only use data generated by the student up to the point where the predictions are generated.

To evaluate the proposed models using data from different sources, not only from Moodle, and to use indicators that are specific to different types of educational environments like blended learning are also possible lines of future work.
The combination of unsupervised learning techniques with supervised learning techniques is another possible line of research. Adding context to indicators based on students activity resulted in an improvement in accuracy. To expand the list of context variables of each course, to encode them in a reduced set of latent features and to append these latent features to students’ activity features could improve the portability of predictive models.

Other machine learning and deep network techniques can also be investigated. Recurrent Neural Networks [55] can process the traces that students leave behind as time series which can potentially lead to improvements in accuracy. Other possible lines of research can explore the timing of the predictions generated by the system and how the prediction models accuracy changes as more data is included.
Appendix A

Indicators for assessment

These are the indicators used in Chapter 4. These indicators are bundled into a Moodle plugin available in https://github.com/dmonllao/moodle-local_assessesindicators under GNU General Public License v3.0.

A.1 Positive ratings in database entries

```php
namespace local_assessesindicators\analytics\indicator;

class received_database_entry_ratings extends \core_analytics\local\indicator\binary {

    /**
     * @var \grade_item
     */
    protected static $gis = [];

    /**
     * Calculates the indicator.
     *
     * @param int $sampleid
     * @param string $sampleorigin
     * @param int $starttime
     * @param int $endtime
     * @return float
     */
    protected function calculate_sample($sampleid, $sampleorigin, $starttime = false, $endtime = false) {
        global $DB, $CFG;

        if (!empty($CFG->disablegradehistory)) {
            return null;
        }
    }
```
$user = $this->retrieve('user', $sampleid);
$course = $this->retrieve('course', $sampleid);
$modinfo = get_fast_modinfo($course);

$ratingfound = false;
$databases = $modinfo->get_instances_of('data');
foreach($databases as $database) {
    if ($database->sectionnum == 0) {
        // Skip top section database activities.
        continue;
    }

    $key = $course->id . '-' . $database->instance;
    if (empty($this::$gis[$key])) {
        $this::$gis[$key] = new grade_item([  
            'courseid' => $course->id,  
            'iteminstance' => $database->instance,  
            'itemmodule' => 'data'
        ]);
    }
}

// We check grade grades history because we probably have a $endtime restriction.
$params = [
    'itemid' => self::$gis[$key]->id,
    'userid' => $user->id,
];
$select = 'itemid = :itemid AND userid = :userid';
if ($starttime) {
    $params['starttime'] = $starttime;
    $select .= ' AND timemodified >= :starttime';
}
if ($endtime) {
    $params['endtime'] = $endtime;
    $select .= ' AND timemodified <= :endtime';
}
if ($DB->record_exists_select('grade_grades_history', $select, $params)) {
    $ratingfound = true;
    break;
}

if ($ratingfound) {
    return self::get_max_value();
} else {
    return self::get_min_value();
}
A.2 Positive ratings in forum posts

namespace local_assessesindicators\analytics\indicator;

class received_forum_post_ratings extends \core_analytics\local\indicator\binary {

    /**
     * @var \grade_item
     */
    protected static $gis = [];

    /**
     * Calculates the indicator
     * @param int $sampleid
     * @param string $sampleorigin
     * @param int $starttime
     * @param int $endtime
     * @return float
     */
    protected function calculate_sample($sampleid, $sampleorigin, $starttime = false, $endtime = false) {
        global $DB, $CFG;

        if (!empty($CFG->disablegradehistory)) {
            return null;
        }

        $user = $this->retrieve('user', $sampleid);
        $course = $this->retrieve('course', $sampleid);
        $modinfo = get_fast_modinfo($course);

        $ratingfound = false;
        $forums = $modinfo->get_instances_of('forum');
        foreach ($forums as $forum) {
            if ($forum->sectionnum == 0) {
                // Skip top section forum activities.
                continue;
            }

            $key = $course->id . '-' . $forum->instance;
            if (empty($this::gis[$key])) {
                self::gis[$key] = new \grade_item(
                    'courseid' => $course->id,
                    'iteminstance' => $forum->instance,
                    'itemmodule' => 'forum'
                );
            }
        }
    }

} 59
APPENDIX A. INDICATORS FOR ASSESSMENT

A.3 The student attempted at least one quiz

namespace local_assessesindicators\analytics\indicator;

class any_quiz_attempt extends \core_analytics\local\indicator\binary {

    /**
     * @var \grade_item
     */
    protected static $gis = [];

    /**
     * Calculates the indicator.
     *
     * @param int $sampleid
     * @param string $sampleorigin
     */
    public function calculate($sampleid, $sampleorigin)
    {
        // We check grade grades history because we probably have a $endtime restriction.
        $params = [
            'itemid' => self::$gis[$key]->id,
            'userid' => $user->id,
        ];
        $select = 'itemid = :itemid AND userid = :userid';
        if ($starttime) {
            $params['starttime'] = $starttime;
            $select .= ' AND timemodified >= :starttime';
        }
        if ($endtime) {
            $params['endtime'] = $endtime;
            $select .= ' AND timemodified <= :endtime';
        }
        if (!DB->record_exists_select('grade_grades_history', $select, $params)) {
            $ratingfound = true;
            break;
        }
        if ($ratingfound) {
            return self::get_max_value();
        }
        return self::get_min_value();
    }
}
protected function calculate_sample($sampleid, $sampleorigin, $starttime = false, $endtime = false) {
    global $DB, $CFG;

    if (!empty($CFG->disablegradehistory)) {
        return null;
    }

    $user = $this->retrieve('user', $sampleid);
    $course = $this->retrieve('course', $sampleid);
    $modinfo = get_fast_modinfo($course);
    $quizzes = $modinfo->get_instances_of('quiz');
    foreach ($quizzes as $quiz) {
        if ($quiz->sectionnum == 0) {
            // Skip top section quiz activities.
            continue;
        }

        $key = $course->id . '-' . $quiz->instance;
        if (!empty(self::$gis[$key])) {
            self::$gis[$key] = new \grade_item([  
                'courseid' => $course->id,
                'iteminstance' => $quiz->instance,
                'itemmodule' => 'quiz'
            ]);
        }
    }

    // We check grade grades history because we probably have a $endtime restriction.
    $params = [
        'itemid' => self::$gis[$key]->id,
        'userid' => $user->id,
    ];
    $select = 'itemid = :itemid AND userid = :userid';
    if ($starttime) {
        $params['starttime'] = $starttime;
        $select .= ' AND timemodified >= :starttime';
    }
    if ($endtime) {
        $params['endtime'] = $endtime;
        $select .= ' AND timemodified <= :endtime';
    }
    $gghs = $DB->get_records_select('grade_grades_history', $select, $params);
    foreach ($gghs as $ggh) {

    }
A.4 Percentage of quizzes attempted in the course

amespace local_latesubmissions\analytics\indicator;

class percent_quiz_attempt extends \core_analytics\local\indicator\linear {

/**
 * Calculates the indicator.
 * @param int $sampleid
 * @param string $sampleorigin
 * @param int $starttime
 * @param int $endtime
 * @return float
 */
protected function calculate_sample($sampleid, $sampleorigin, $starttime = false, $endtime = false) {
    global $DB;

    $course = $this->retrieve('course', $sampleid);
    $coursemodule = $this->retrieve('course_modules', $sampleid);
    $user = $this->retrieve('user', $sampleid);

    if (!$logstore = \core_analytics\manager::get_analytics_logstore()) {
        throw new \coding_exception('No available log stores');
    }

    $quizzes = [];
    $modinfo = get_fast_modinfo($course, $user->id);
    foreach ($modinfo->get_instances_of('quiz') as $cm) {
        if (!$cm->uservisible) {
            continue;
        }

        $quizzes[] = $cm;
    }

    $smartname = $this->get_name($sampleid, $sampleorigin, $course->id, $coursemodule->id, $user->id);
    $smartname = $this->get_name($sampleid, $sampleorigin, $course->id, $coursemodule->id, $user->id);
    $smartname = $this->get_name($sampleid, $sampleorigin, $course->id, $coursemodule->id, $user->id);
    $smartname = $this->get_name($sampleid, $sampleorigin, $course->id, $coursemodule->id, $user->id);

    return $this->calculate($course, $coursemodule, $user, $quizzes, $smartname);
}
}
A.5. THE STUDENT FAILED A QUIZ ATTEMPT

continue;
}
if (empty($this->cms[$cm->id])) {
    $this->cms[$cm->id] = $DB->get_record($cm->modname, array('id' => $cm->instance));
}
$quizzes[$cm->id] = $cm;
if (!$quizzes) {
    return null;
}
// Number of attempts indexed by context.
$attempts = [];
list($ctxsql, $ctxparams) = $DB->get_in_or_equal(array_keys($quizzes), SQL_PARAMS_NAMED);
$select = "userid = :userid AND contextlevel = :contextlevel AND contextinstanceid $ctxsql AND ".
    "crud = :crud AND eventname = :eventname";
$params = array('userid' => $user->id, 'contextlevel' => CONTEXT_MODULE, 'crud' => 'u', 'eventname' => \mod_quiz\event\attempt_submitted') + $ctxparams;
// We don’t expect much logs so we can afford to retrieve them all.
$logsql = $logstore->get_events_select($select, $params, 'timecreated ASC', 0, 0);
foreach ($logs as $log) {
    $attempts[$log->contextinstanceid] = true;
}
return ((count($attempts) / count($quizzes)) * 2) - 1;
}

A.5 The student failed a quiz attempt

namespace local_assessesindicators\analytics\indicator;
class any_failed_quiz_attempt extends \core_analytics\local\indicator\binary {

/**
 * @var \grade_item
 */
protected static $gis = [];

/**
 * Calculates the indicator.
 * @param int $sampleid
 * @param string $sampleorigin
 */
protected function calculate_sample($sampleid, $sampleorigin, $starttime = false, $endtime = false) {
    global $DB, $CFG;

    if (!empty($CFG->disablegradehistory)) {
        return null;
    }

    $user = $this->retrieve('user', $sampleid);
    $course = $this->retrieve('course', $sampleid);
    $modinfo = get_fast_modinfo($course);
    $failedquiz = false;
    $anyattempt = false;
    $quizzes = $modinfo->get_instances_of('quiz');
    foreach ($quizzes as $quiz) {
        if ($quiz->sectionnum == 0) {
            // Skip top section quiz activities.
            continue;
        }

        $key = $course->id . '-' . $quiz->instance;
        if (empty(self::$gis[$key])) {
            self::$gis[$key] = new \ grade_item([
                'courseid' => $course->id,
                'iteminstance' => $quiz->instance,
                'itemmodule' => 'quiz'
            ]);
        }
    }

    // We check grade grades history because we probably have a $endtime restriction.
    $params = [
        'itemid' => self::$gis[$key]->id,
        'userid' => $user->id,
    ];
    $select = 'itemid = :itemid AND userid = :userid';
    if ($starttime) {
        $params['starttime'] = $starttime;
        $select .= ' AND timemodified >= :starttime';
    }
    if ($endtime) {
        $params['endtime'] = $endtime;
        $select .= ' AND timemodified <= :endtime';
    }
}
A.5. THE STUDENT FAILED A QUIZ ATTEMPT

```php
$gghs = $DB->get_records_select('grade_grades_history', $select, $params);
foreach ($gghs as $ggh) {
    $ggh->rawgrademin = floatval($ggh->rawgrademin);
    $ggh->rawgrademax = floatval($ggh->rawgrademax);
    $ggh->finalgrade = floatval($ggh->finalgrade);
    if (!$ggh->finalgrade) {
        // Discards 0.000 and nulls.
        continue;
    }
    if (empty($anyattempt)) {
        $anyattempt = true;
    }
    if (floatval(self::$gis[$key]->gradepass) > 0) {
        $gradepass = self::$gis[$key]->gradepass;
    } else {
        // (Max - min mean) + min.
        $gradepass = (($ggh->rawgrademax - $ggh->rawgrademin) / 2) + $ggh->rawgrademin;
    }
    if ($ggh->finalgrade < $gradepass) {
        $failedquiz = true;
        break 2;
    }
}
if ($anyattempt === false) {
    // We separate users without quiz attempts from users without fails.
    return null;
}
if ($failedquiz === true) {
    return self::get_max_value();
} return self::get_min_value();
```
Bibliography


List of Figures

3.1 A Data flow diagram showing the steps that the framework goes through in the production mode. It includes examples of each component. .............................................. 11
3.2 The class diagram of the framework. ................................................................. 12
3.3 Test results. ........................................................................................................ 19
3.4 Actionable insights generated by an at-risk students model. .............................. 20
5.1 Students’ activity contextualised through training. ........................................... 44
5.2 Students’ activity contextualised through training alongside the original SA inputs. . 45
5.3 Students’ activity and context trained separately. ............................................. 46
5.4 Students’ activity and context trained separately with one context hidden unit. .... 46
5.5 Students’ activity and context trained separately, considering peers activity as part of the students’ activity. .............................................................. 46
5.6 Students’ activity and context trained separately. ............................................. 47

List of Tables

3.1 Prediction model evaluation results ................................................................. 19
4.1 Prediction model evaluation results ................................................................. 31
4.2 Indicators’ correlation to drop-outs. ............................................................... 31
5.1 Datasets used for this study ........................................................................... 35
5.2 Using unseen datasets as testing data (one-size-fits-all). Average accuracy and F1 score after running 1000 epochs and 5 repetitions. ................................................. 49
5.3 Splitting each dataset in training and testing. Average accuracy and F1 score after running 1000 epochs and 5 repetitions. ................................................. 49