Education as a Complex System

An investigation of students' learning behaviours in Programming Education using Complexity approaches.

Tai Tan Mai, BSc, MSc

Supervised by Prof. Martin Crane and Dr. Marija Bezbradica

A thesis presented for the degree of Doctor of Philosophy

School of Computing

Dublin City University

Date: August 2022

Declaration

I hereby certify that this material, which I now submit for assessment on the programme of study leading to the award of Doctor of Philosophy is entirely my own work, and that I have exercised reasonable care to ensure that the work is original, and does not to the best of my knowledge breach any law of copyright, and has not been taken from the work of others save and to the extent that such work has been cited and acknowledged within the text of my work.

Date: 28/08/2022 **Signature of the Candidate**

ID No: 17213604

Acknowledgement

This research is financially supported by the Irish Research Council Governmental Postgraduate Scholarship Scheme under the project number GOIPG/2017/141, also affiliated with the ADAPT Centre for Digital Content Technology at the School of Computing, Dublin City University.

During my PhD study, I have received a great deal of support and assistance. I would like to thank the following people for helping with this research project:

Prof. Martin Crane and Dr. Marija Bezbradica, my supervisors, for a wonderful guidance during my PhD journey. Their expertise, experience and patience are invaluable in directing me going the right way, especially when I faced big issues about the departure of my ex-supervisor 2.5 years ago, and then I had to switch my research domain and then the pandemic occurred. I believe that what they have done for me is far beyond the ordinary responsibilities of a PhD supervisor, which brought my work to a higher level.

Dr. Stephen Blott for his kind support on data collection and using the Einstein learning system at the School of Computing.

Prof. Markus Helfert and Dr. Thoa Pham for their encouragements and help in the preparation for my PhD funding application as well as the support during the first two years of my study.

Professors, administrative staff and PhD researchers at the School of Computing, DCU, for creating a great research environment and network.

My Vietnamese, Irish and all friends in Ireland for many great parties with good food, great conversations and nice music!

My parents for their unconditional support, encouragement and patience over years as always!

Last but not least, Dao, my dearest wife, for everything but most especially for her willingness to join my journey in Ireland despite many changes in life that she has to overcome. Her appearance made this journey far less difficult and even more meaningful to me, and to us!

Contents

List of Figures

List of Tables

List of Abbreviations

RMT	Random Matrix Theory
EPM	Educational Process Mining
LMS	Learning Management Systems
LCMS	Learning Content Management Systems
MOOCs	Massive Open Online Courses
RNNs	Recurrent Neural Networks
LSTM	Long Short-Term Memory
PCA	Principal Component Analysis
PC	Principal Component
CGPA	Cumulative Grade Point Average
MST	Minimum Spanning Tree
RQ	Research Question
CS	Computer Science
IT	Information Technology
STEM	Science, Technology, Engineering, Mathematics
IPR	Inverse Participation Ratio
EDM	Educational Data Mining
LA	Learning Analytics

List of Publications and Presentations during the PhD study

Peer-reviewed Journal Publications

- Mai, T. T., Bezbradica, M., & Crane, M. (2022). Learning behaviours data in programming education: Community analysis and outcome prediction with cleaned data. Future Generation Computer Systems, 127, 42-55.

Peer-reviewed Conference Publications

- Mai, T. T., Crane, M., & Bezbradica, M. (2021, July). Students' Behaviours in using Learning Resources in Higher Education: How do behaviours reflect success in Programming Education?. In Proceedings of the 7th International Conference on Higher Education Advances (HEAd'21) (pp. 47-55).

- Mai, T., Pham, T., & Helfert, M. (2019). An investigation of discovering business processes from operational databases. In Proceedings of the 24th UK Academy for Information Systems conference, 2019, Oxford, The UK.

Other Presentations

- Mai, T., Bezbradica, M and Crane, M. Education as a complex system: using physic methods for the analysis of students' learning behaviours in the context of programming education. In 11st Polish Symposium on Physics in Economy and Social Sciences, 1-3 July 2021, Krakow, Poland.

- Yang, Qishan and Mai, Tai Tan. Reviewing features of Data Warehouse Architectures: a taxonomy. In Digital Business Research Day (DBRD) 2018, 11 June 2018, Linz, Austria.

Abstract

Education as a Complex System:
An investigation of students' learning behaviours in
Programming Education using Complexity approaches.
Tai Tan Mai

As a result of the COVID19 pandemic, more higher-level education courses have moved to online channels, raising challenges among educators in monitoring students' learning progress. Thanks to the development of learning technologies, learning behaviours can be recorded at a more fine-grain level of detail, which can then be further analysed. Inspired by the premise of approaching education as a complex system, this research aims to develop a novel approach to analyse students' learning behavioural data in programming education, utilising complexity methods. First, essential learning behavioural features are extracted. Second, a novel method based on Random Matrix Theory is developed to remove the noise and trend effect in the data in order to better highlight the differences in students' learning behaviours. Third, Community Detection is applied to cluster the students into groups with similar learning behavioural characteristics. In the thesis also, motivated by a need to determine likely outcomes of students, a range of machine learning classification techniques have also been applied to predict the student learning outcomes based on behavioural data which has been cleaned of the noise and trend.

The proposed approaches have been applied to datasets collected from a bespoke online learning platform in an Irish University. The datasets contain information from 566 students in different programming-related modules over a range of years encompassing pre and during the COVID19 pandemic. This gives us a unique opportunity to test our methods for the effects of the pandemic on learning. Results indicate the similarities and deviation in learning behaviours between student cohorts.

Overall, we found that students interacted similarly with all course resources during the semester. However, while higher-performing students seem to be more active in practical tasks (e.g. programming exercises on labs), lower-performing students have been found to focus overmuch on lecture notes and lose their focus at the later phase of the semester. Additionally, students' learning behaviours in a conventional university setting tend to differ significantly to those students in a fully online setting during the pandemic. We have also attempted to reduce the noise component in the data and the experimental results further demonstrate the better prediction performance of models which are trained based on the cleaned dataset, in comparison with the original dataset. Recommendations for current educational practice are made, including the continuous analysis of learning behaviours by the proposed methods and suggestions for the prompt interventions to in order to maximise student supports.

Chapter 1

Introduction

1.1 Background and Motivation

Education in Computer Programming with related domains has received increasing attention due to the growth in demand for Information Technology (IT)-related job markets. Furthermore, in recent years, STEM fields (Science, Technology, Engineering and Mathematics) also have the demand for essential IT skills and knowledge. This makes these types of skills an integral part of many STEM sub-disciplines such as Artificial Intelligence, Bio-informatics, Statistics. One of the pivotal and essential elements in any IT-related degree is a suite of programming courses so. As a result, understanding and improving students' engagement and process of learning on the learning materials are of key importance for the best practice in pedagogy [110]. However, despite the growing demand for these type of professionals, there have been relatively high dropout rates in introductory programming courses reported from many studies [23]. The failure rates in introductory programming modules has been reported to be 28% on average, with a huge variation from 0% to 91% [23], according to a recent study using data from 161 universities around the world.

Online learning environments (e.g. online programmes, Massive Open Online Courses - MOOCs), which are usually supported by Learning Management Systems (LMS), have been commonly used in higher education. While online learning has

shown its benefits [65], it still contains drawbacks for higher education [74]. Students often participate in online educational programmes mainly as solitary learners with a private account and complete learning tasks on an individual basis. Hence, the ability to engage in self-directed learning plays an important role in the success of the students [191], especially in an online learning setting. A student needs to be aware of how to use learning tools and materials properly to achieve learning outcomes, which are usually determined through performance on both formative and summative assessment tasks. Such formative assessment tasks are often used to reflect the mastery of the course materials [73]. However, getting negative results in assessment can further impact student's performance as it can affect students' confidence, which may lead to their disengagement [157]. Therefore, monitoring and predicting students' performance could provide an opportunity for developing an *adaptive* learning environment which is responsive to students' academic progress, i.e. providing students with recommendations and feedback based on their learning behaviours to maximise the likelihood of student success in the learning process.

Typically, lecturers can monitor student progress and, if necessary, conduct interventions to ensure the quality of learning and progress of programming students [164]. In addition to practical exercises and formal assessments, interaction in conventional face-to-face classes can help educators understand students' performance. However, restrictions in face-to-face classes, e.g. in online or hybrid courses, can restrict the potential for direct communication between educators and students [107]. These difficulties may create additional challenges for lecturers in monitoring how students are performing during the courses. For example, a fully online or open-book/open-note exam could negatively influence the monitoring of cheating [12]. Additionally, more higher education programmes have moved to online and hybrid channels due to the pandemic, further exacerbating the lack of direct communication. Hence, it is necessary to develop novel methods to support educators in monitoring and understanding students' learning behaviour during their online sessions.

The development of Learning Management Systems (LMS)[1] and Learning Content Management Systems (LCMS)[2] have enabled the capturing of learning data generated by participants of the courses [166]. These systems provide the ability to automatically record a large amount of interaction data at fine-grained levels, e.g. at the level of mouse and keyboard events on a screen. Such log data can be used by educators to improve the pedagogical value of teaching and learning [145]. Analysing the massive amount of educational data thus collected during the learning process also has the potential to help instructors and students to obtain a comprehensive view of a student's learning progress. As a result, these data could provide valuable insights into individual students' ongoing learning progress and thus improve the learning environment, as has been suggested in [16]. Data mining can help to identify a group of emerging patterns - e.g. discovering a set of study patterns of different student cohorts and the correlation between the patterns and students' learning performance. These characteristics can be used to estimate the possibilities of success or failure of students in the courses. Such insights enable evidence-based interventions and recommendations [90], which can lead to significant improvement in students' learning outcomes [60].

1.2 Research problems and questions

In relation to behavioural log data, there is the potential for the effect of *noise* and *trend* to be present in the automatically collected data. This arises from students working flexibly when completing their learning paths in the online learning system. For example, they can carry out various learning activities such as reading lecture notes, coding, navigating among course documents in any order, resulting in noise in the logged data, i.e. data heterogeneity and complexity [72]. In addition, we have

[1] An LMS is software that automates the training process and function and includes registration and administration tools, skills and records management, courseware access, and programming interfaces to packaged applications ([70]).

[2] An LCMS is an integrated set of technology that manages all aspects of learning content. This includes authoring or acquisition, content history, auditing, replacement, and deletion. An LCMS generally works in conjunction with an LMS ([70]).

noticed from the data gathered that students are likely given the same instructions and learning pathway in the same class. As a consequence, this may create a *trend* effect, i.e. students' learning behaviours can be similarly and highly positively correlated with other learners' behaviours in the same course. Hence, as an initial task it is important to process the data, i.e., to filter noise and clean the trend effect in the event log data before applying further analysis.

This research aims to investigate the relationship between students' learning behaviours on course material items and their performance in the formative assessments while taking programming-related courses. Specifically, the research objective is to answer the following research questions:

Research question: Is there a relationship between student learning behaviours and student learning performance in programming education?

- Sub RQ1: How can learning behaviours be captured and processed to support further, more detailed, analysis?
- Sub RQ2: Do students from different groups, corresponding to different patterns of learning behaviours, perform differently in the exams? If this is the case, how the use of course material items might differ among the student groups?
- Sub RQ3: Do student learning behaviours differ before and during the COVID19 pandemic in terms of interacting with course material items?
- Sub RQ4: Could students' learning behavioural data be used to predict learning outcomes (Pass or Fail) in the early stages of the study period using results from formative assessments?

In this work we adopt the terminology from [167] for the terms *programme* and *course*. In particular, a *programme* refers to a series of courses that a student needs to complete to obtain a university degree. A *course* is a building block or subject of a programme, containing a set of learning components, such as lectures, tutorials, lab practice sessions and assessments.

To address the research questions in this thesis, we utilise the dataset which contains the data for 566 university students participating over the two programming-related courses, namely Course#1 and Course#2, during the three academic years before and during the COVID19 pandemic, i.e. the pre-COVID19 data 2018 and 2019, and the during COVID19 dataset in 2020. The pre-COVID19 courses have been delivered to students in a combination of conventional and online formats. In particular, students have physically attended the lecture sessions in lecture halls and have conducted all learning activities on a bespoke online system. The learning data were logged and these serve as input datasets for our further analysis. Behavioural data captured automatically from the system is stored in the format of *event log*. During the pandemic closure, students have studied at home, and all were expected to have the same accessibility to the system. From the input event logs, the concept of a *student-event data matrix* and a *transition-student data matrix* (described below) have been developed to represent the students' learning behaviours. To deal with the problem of noise and trend effect in the datasets, we utilise a cleaning method based on Random Matrix Theory (RMT), followed by the construction of Minimum Spanning Trees (MST) to reflect the difference in learning behaviours of all students. Community Detection algorithms and statistical tests have also been applied to investigate the students' behaviours on course material items. For the prediction of student learning outcomes, a set of machine learning algorithms have been applied on every week's original as well as cleaned data and the predictability has been validated by cross-validation technique.

The following chapters of this thesis are structured as follows. Chapter 2 discusses the existing approaches in learning behavioural analytics. Chapter 3 introduces the context of the study through the collected datasets and explanatory analysis. Chapter 4 describes the research methodology, including data collection and processing as well as data analysis methods. In this thesis, we utilise complex systems methods, i.e, Random Matrix Theory and Community Detection, in combination with other data mining techniques, consisting of Principal Component Analysis and classification techniques. Chapter 5 shows the experimental results of

the analysis, followed by Chapter 6 which discusses the implications and limitations of the research results, as well as concludes the thesis and briefly introduces the possible future work in line with this research.

Chapter 2

Literature Review

2.1 Introduction

With the availability of educational data, the use of data mining techniques has opened up a great potential to find solutions for many specific problems in higher education institutions [173]. Learning Analytics (LA) and Educational Data Mining (EDM) can be seen as the two specific domains which refer to the applications of data mining in education. It has been shown that they can help to create more adaptive and interactive educational environments, and can therefore improve teaching and learning quality both for educators and students alike [124]. Applications of data mining tools and techniques in education have various different aims, from student behaviour modelling to prediction of performance and enhancement of assessment feedback as well as improvement in student learning outcomes[130].

Similar to any data mining application, an EDM/LA project is generally carried out through three major phases: data preparation, analysis and post-processing [42], as can be seen in the diagram in Figure 2.1.

- **Data preparation:** Data preparation includes data collection and pre-processing, which is the first step of any EDM/LA process [82]. This step plays an important role in the success of an EDM/LA project. Where the collected data is not yet ready for further analysis, a number of data pre-processing tasks

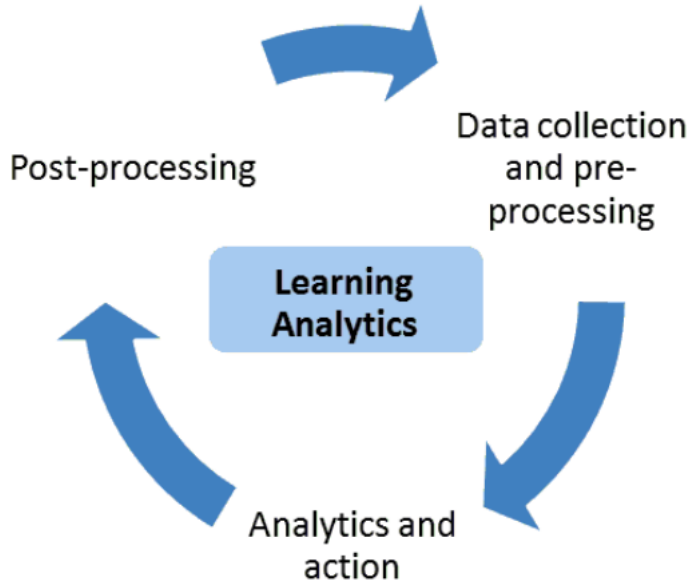


Figure 2.1: Phases of a EDM/LA process [42]

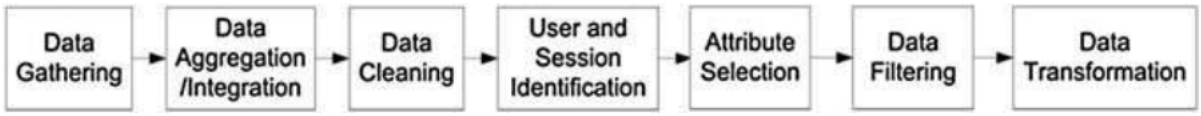


Figure 2.2: Example of main steps/tasks in pre-processing of educational data [147]

need to be conducted in this phase, for instance, data cleaning, data filtering and data transformation [147]. Figure 2.2 shows a diagram of pre-processing phases for educational data.

- **Analysis:** Using the collected and pre-processed educational data, various data analytics techniques can be applied to discover underlying insights and valuable patterns from the learning and teaching experiences. The selection of the data mining techniques is subject to the research objectives (e.g. prediction of students' learning outcomes, identifying "at-risk" students) and characteristics of collected data (e.g. categorical, time series, continuous data) [42].

- **Post-processing and improvement:** If necessary, the results of the analysis phases may need to be continuously improved due to practical demands. This phase, therefore, can involve collecting new data, selecting new attributes,

using new metrics and applying additional analysis algorithms. Generally, one can see an EDM/LA application as an iteration of data collection, pre-processing and analysis on educational data [42].

This chapter covers two key aspects of EDM/LA research: First, we summarise the aspect of data, focusing on how existing research in EDM/LA collect and pre-process different type of educational data; Second, we discuss various approaches for the data analysis of learning behaviours, consisting of traditional data mining approaches and educational process mining. In addition, the main applications related to learning behavioural analytics have also been investigated.

2.2 Data in Educational Data Mining

2.2.1 Data formats for Educational Data Mining

Students' learning behaviours refer to the learning activities that students have participated in or are involved with during the period of the course. The learning behaviours of the learners in a course can be captured in different ways, subject to the course's educational setting. For example, such behaviours can be reflected through educators' records on students' activities such as attendance, reading lecture notes and homework exercises completion, which are also called "engagement levels" [60]. The level of detail of such records varies, depending on which type of data the supported system can log into the database. The main types of data for this domain can be listed as follows [147]:

Transaction/tabular data:

A transaction dataset refers to a table where each column represents an attribute/variable and each record/row includes a list of data values associated with their attributes. Data attributes can also refer to data features that need to be determined before the analysis step [82]. The selection of data features is also subject to analysis purposes. In the context of EDM, each data row can represent the feature values of a student.

Data feature values can be automatically recorded or derived/aggregated from other data sources. For example, in [48], the authors use the Moodle administration functions to extract data related to each student such as 'number of messages read', 'number of active days' and 'number of assignments handed via the platform'. Tabular data can be seen as one of the most commonly used data type as it is necessary input for a large number of data mining methods, such as classification and clustering [147], for example, Supported Vector Machine, Random Forest and K-means clustering.

Relational data:

A relational database is a collection of tables that link to each other through a data relationship (e.g. *one to many*, *one to one*). Each table is made up of a set of attributes (columns or fields) and typically stores a large number of records (or rows) [149]. Data in a relational database can be extracted by database queries that are written in a query language such as Structured Query Language (SQL) or with the support of designated user interfaces. Although there are relational data mining techniques, in practice, relational data are normally transformed into transaction data to reduce complexity [149].

Temporal data:

A temporal dataset can be seen as a special tabular dataset, with sequences of events over time [82]. The data features may involve the name of the learning event (e.g. students' action on an activity such as *open lecture notes*, *upload assignment* etc.), time-related attributes (e.g. timestamp, starting time, ending time), or identities (e.g. IP, student number, location). This type of data is also a common option and requires special data mining techniques which are specifically designed to discover sequential patterns, such as time series, sequential data mining and process mining methods. Additionally, temporal data can be transformed into tabular data to suit for other common data mining algorithms. For example, temporal data can be

stored in form of an *event log* which is designed for process mining algorithms [174, 151].

Text data:

Text data consists of a large volume of documents, which can be collected from various sources. Text data can be highly unstructured, e.g., essays, chat messages and web pages, or semi-structured e.g., e-mail messages and forum posts. Some courses may contain special documents, e.g. student submitted code in a programming course. Mining unstructured text data may refer to the domains such as "Text Analytics" [121] and "Natural language processing" [47]. Such text data mining applications have demonstrated their usefulness in educational contexts, e.g. assessing the students' usage of asynchronous discussion forum [59], which can be used for predicting students' academic performance [182].

2.2.2 Static and Dynamic data in learning management system

In general, input data in relation to educational prediction can be classified into two categories: *static* and *dynamic* data [60]. Student demographic information and historical educational records can be classified as static data because these variables and values do not update or change frequently over the study period. On the other hand, online behaviours, textual data and other multi-modal data can be considered as dynamic data because they can be continuously generated when students interact with the system. In terms of the prediction of learning outcomes from the static data, one can rely on features such as personal attributes and cumulative grade point average (CGPA) from previous years [10]. For instance, using learning grades from previous courses, e.g. programming or mathematics, can help in predicting the drop-out probabilities of computing students [64]. However, the use of static data to predict learning outcomes has been shown to cause some problems [60], i.e. the student's actual efforts during the learning process can be ignored. It has also been

found that previous student results (e.g. the CGPA) are not sufficient to predict tendency to drop-out, and that engagement variables also need to be included (e.g., number of accesses to the platform) to achieve good accuracy results [5]. The static data can also be difficult to collect as they may need to be merged from various data sources, which might affect data quality and cause ethical issues.

On the other hand, dynamic features, such as behavioural data, can be collected easily due to their availability in advanced learning platforms. In this way, it is possible for one to take advantage of dynamic data [53] to predict student learning outcomes. For example, learning log data from the Moodle[1] platform has been used for predicting learners' performance [71], using common features in a Learning Management System such as Assignment, Feedback, Course login and Chat. More fine-grained data can be used as predictors such as mouse interactions (e.g. *click* and *drag*) [176]. Multi-modal features (e.g. eye-tracking, face-video and wristband) have also demonstrated the potential for predicting learning performance [161]. Generally, according to the recent surveys, most of the current approaches have a focus on either combining new features collected from learning platforms or new strategies with different machine learning predictive models [115, 60].

2.2.3 Data pre-processing in EDM/LA

Data pre-processing relates to a set of activities that may need to be carried out to ensure that the input data is appropriate with the analysis methods. These activities may include *data gathering*, *data cleaning*, *data filtering*, *data transformation* and *data format conversion* [147].

Data Cleaning

Data cleaning refers to the process of detecting unwanted phenomena in the dataset and discarding irrelevant data before further analysis [82]. The common types of

[1]Moodle is an open-source learning management system (LMS) and distributed under the GNU General Public License. Moodle is developed in PHP and can be seen to be one of the most commonly-used LMS platforms

such errors in EDA/LA have been shown to be missing data and inconsistent data [86]. Inconsistent data occur when data values have been found to be inconsistent with other relevant values. On the other hand, missing data refers to the phenomenon where there is no value stored for the attribute(s) in certain records [101]. In educational data, missing data values may occur due to several reasons, such as students not completing all required learning tasks or being absent in some learning sessions as well as technical errors during the data collection progress. Inconsistent and missing data issues can be solved by mean of different strategies, e.g. imputation methods which aim to replace missing values with substituted values (e.g. mean or median) as well as more sophisticated approaches such as regressions [123] or association rules [7]. In some cases, one can completely remove observations that contain missing values, e.g. the data for students who only attended the first session but never showed up again can be removed.

In addition, one of the most difficult issues in data pre-processing is dealing with noisy data [86, 147]. Existing research efforts in EDM/LA have considered removing noisy data from the dataset, for the simple reason that noisy data have been generally seen as outliers, i.e. samples in a larger dataset that do not comply with the general behaviour of the data [147, 93]. The outliers are significantly different from the remaining of the data distribution, which may be caused by the error of data collection, unexpected or unlikely behaviours within the system. To overcome such issues, data mining techniques can be used, e.g. one may use median instead of mean values [19], to minimise the effect of outliers in the constructed models. However, in education, noise in the data can also have true observations. For example, a few students in the class can achieve outstanding final grades with little effort, i.e. they may be recorded to have only a few interactions with exercise items in comparison with most of the average successful students. One may require domain knowledge and analysis purposes to detect such outliers or noise data [139]. In our work, rather than removing the noise, we concentrate on extracting useful information from noise.

Data Filtering

This process aims to select a subset of data from the original dataset so that the input data can be manageable within the constraints related to computational power and analysis method [82]. This process can help to significantly reduce the amount of data and unnecessary information. For example, the whole original dataset may contain a hundred features and many of them may be irrelevant to the analysis purposes. A simple approach to data filtering is that the researchers define the inclusive conditions and exclude all data that violate them [120], e.g. data can be filtered by academic years, courses, semesters, activities. More sophisticated methods can also be applied to filter the data, such as was used in [165] where the authors proposed the use of activity theory to map the original educational data to a higher-level representation.

On the other hand, fine-grained data collected from a learning management system can show different levels of granularity, such as keystroke and mouse click level, session-level, student level, classroom level etc [15]. Students' learning behaviours can be reflected by excessively detailed learning logs which contain students' interactions with course material items at the level of mouse clicks on every relevant web page of the course [58, 72]. For example, when a student clicks on a keyword in an online lecture note of the course, the action of clicking will be stored in the database, along with relevant attributes e.g. student identity, timestamps, the keyword etc. Working with such a behavioural dataset, one may define different levels of abstractions depending on the scope of the analysis [147]:

- **Event:** a single action or interaction event (e.g mouse clicking and scrolling, key pressing);
- **Activity:** a sequence of one or many events to achieve a particular outcome;
- **Session:** a sequence of events generated by a user since his/her login until the last interaction;
- **Task:** a sequence of events of a user within a single course resource item.

Higher grained data may include a smaller volume of data and vice-versa. Therefore, it would be important to identify the level of granularity from the collected data.

Data transformation

Data transformation refers to the activity of constructing secondary attributes from available attributes [82], which can provide better interpretability of information. Several commonly-used transformation techniques namely normalisation, discretisation, and derivation are described as follows:

- **Normalisation** is a technique where the attribute values are re-scaled within a specific value range, e.g. from -1.0 to 1.0 or from 0 to 1. Usually, normalisation can be used when we want to re-scale the magnitudes of data values to low values, which may improve the accuracy and efficiency of mining algorithms by involving distance measurements [82] such as clustering techniques [102].

- **Discretisation** splits the numerical data values of an attribute into categorical values that may be more comprehensible than the original ranges and magnitudes due to the lower number of possible values. This technique would be helpful when the selected analysis method does not work effectively with continuous data, e.g. association rule. Discretisation needs the rule to classify original values. For example, with the "equal-width binning" technique, the range of possible new values will be automatically divided into N sub-range of the same size, where N bins is a pre-defined parameter. The division can also be done manually where users directly specify the cut-off points for each categorical value. It can be seen in the educational context, for instance, with the classification of students' overall results based on marks or grades.

- **Derivation** refers to the activity of generating new features from the current attributes. A new attribute can be derived from one or more attributes available in the dataset through some kind of aggregation or mathematical transformation.

2.3 Methods for the analysis and prediction of learning behaviours

A diversity of data mining techniques has been applied to historical educational data to discover valuable insights in students' learning progress [16, 148, 147, 28]. The sections below summarise some of the most commonly applied techniques that have been reported in the recent reviews [8, 143].

2.3.1 Data Mining techniques in Education

Classification, which aims to divide observations into different pre-defined categories based on data attributes, can be seen as one of the most commonly-used data mining techniques in higher education [8]. In the context of education, classification techniques can be used for several targets such as predicting students' learning performance/achievements and detecting problematic behaviours, to help in the enhancement of teaching and learning quality. For example, among existing studies, a Decision Tree approach was used to accurately predict students' final grades in a C programming module [136]. Student skills in the Java programming language were also modelled by the Additive Factors algorithm in [190]. Here, the authors of the latter study built a model that has been reported to be accurate for both modelling student knowledge and suggesting the concepts that students can address in their code. In a similar study, Bayesian algorithms were applied to model students' knowledge over time based on historical quiz responses [126]. The author reported the possibility of predicting, with good accuracy, whether a student has acquired the relevant knowledge during the course, i.e., the prediction of the students' correct answers in the quizzes.

Clustering refers to the set of techniques aiming to identify groups/clusters/patterns of data that show similarities in their features and establish useful inferences [146]. For example, one may find groups of students who appear to have similar learning characteristics based on visiting content on the webpages of the LMS, which can be

used to recommend preferred learning activities and resources to students [13, 81]. Additionally, behaviour-related attributes, e.g. total time spent on theoretical and practical contents and forums, can be used to detect clusters of procrastination and thus to focus students, e.g. by setting intermediate time goals [41]. The authors in [68, 141] created a network structure of undergraduate courses and applied Community Detection algorithms to identify the contributions of the courses to students' learning pathways. Such findings can support the understanding of students' behaviours in various learning situations [163] and the identification potential dropouts at the early stage of the academic year [41].

Neural networks have also been demonstrated as an effective tool for prediction of students' performance [104]. One can, for instance, use Recurrent Neural Networks (RNNs) to model students' knowledge based on their performance on previously answered quizzes [132]. When a student begins a quiz, the authors predict whether the quiz would be answered correctly. Historic sequences of (*correct* and *incorrect*) quiz responses determine the acquisition of knowledge for that student. Then, the authors create a graph of conditional influence between quiz concepts that can be used for curriculum design. For example, concepts that are highly connected can be grouped in the same module section. On top of that, the effectiveness of using RNNs in the prediction of learning outcomes was confirmed in a later study. The authors utilised an RNN for students with different learning abilities separately [114]. In general, these approaches focus on modelling students purely based on content learned. Beyond the learning content and past performance, it would be also critical for students to adopt good learning behaviours, i.e. skills in accessing and working with relevant content in an appropriate order. In terms of the effect of the learning pattern to predict the learner outcome, one of the most recent pieces of research has indicated the effect of learning activities on the prediction of students' outcomes in MOOCs using Multi-layer Neural Network (MNN) and Long Short-Term Memory Neural Network (LSTM) models [106]. However, the authors merely see the learning behaviours as a single sequence of activities rather than a process that contain an ordered set of sequences, i.e. process cases or traces.

On the other hand, there is a scarcity of research on analysis between the predictability of students' learning patterns using cognitive and behavioural factors and their exam performance [159]. Such analysis can be helpful to provide students with up-to-date suggestions to maximise the likelihood of their learning success.

2.3.2 Educational Process Mining and its applications in Education

Complementing the works in Education Data Mining mentioned above, the emergence of Educational Process Mining (EPM) has expanded the literature on the analysis of massive learning behavioural log datasets, alongside the application of process mining [174] techniques and algorithms in educational event log data [151]. Process mining aims to discover, monitor and enhance processes by extracting information from input data (the so-called "event-log") [4]. Process discovery is one of the most popular techniques in process mining [3]. Process discovery techniques are used to mine an event log to produce a process model. Several algorithms have been developed to discover the process model. Some of the most popular algorithms that have been used in Educational Process Mining (EPM) are Heuristic miner[178, 177], Fuzzy miner[79], Alpha Miner [2], and Genetic mining [109]. The basic idea of the state-of-the-art process discovery techniques is to detect the real process based on behaviours recorded in event logs. Then, the discovered process can be visualised by popular notations for process modelling such as Petri-net[1], Business Process Modelling Notation (BPMN) [122], Fuzzy net, Heuristic nets [79], amongst others.

Recently, EPM has proved to be an effective tool in analysing educational data and delivering new insights into the learning and teaching processes. Many of the process mining applications in education have been developed and implemented in various aspects of education [140, 58]. The prevalence of Massive Open Online Courses (MOOCs) and LMSs have attracted the majority of researchers due to the availability of input log data [72]. The majority of applications in EPM aim to discover learning patterns from the *event log*, resulting in learning process models. The

process models can then be compared according to characteristics of a model such as the number of nodes, edges, split and join points, and transitions. For example, by comparing the learning patterns between the higher and lower performing student groups in practical learning sessions, deviations between those two groups have been revealed [156]. However, there may be challenges related to the construction of process models when there are many complicated and noisy event logs. In such cases, the process mining techniques would likely produce 'spaghetti-like' process models where there is a high density of connections between the model's nodes, which can be incomprehensible [29] and difficult to interpret the models' characteristics [36].

2.3.3 Random Matrix Theory

Random Matrix Theory (RMT) aims to provide an understanding into the characteristics (e.g., statistics of matrix eigenvalues) of matrices whose entries are drawn randomly. RMT was first applied to study the energy levels of complex quantum systems [180, 62]. Deviations from the universal predictions of RMT were then used to identify system-specific, non-random properties of the system under consideration, providing findings of the underlying interactions [111]. RMT has also been applied in many other fields, including wireless communications [172], number theory [113], financial and other large dimensional data analysis [50] financial time series analysis [49], as well as in multivariate statistical analysis and principal components analysis [46]. Recently, RMT has been applied to analyse the price changes of different cryptocurrencies [43].

In general, applications using the Random Matrix Theory approach analyse the properties of the correlation matrix $\mathbf{C}$ of a dataset, to investigate if a large proportion of eigenvalues of $\mathbf{C}$ agrees with RMT predictions. For example, in the context of educational data, the correlation matrix may refer to the similarities between the interactions of students in a course. This finding indicates a considerable degree of randomness in the measured correlations. In addition, deviations from RMT predictions are also used to find the main characteristics, or "information part", of the

dataset. To the best of our knowledge, there have been, to date, no applications of RMT in either Process Mining or educational domains. In this thesis, we investigate the use of RMT applied to a cross-correlation matrix $\mathbf{C}$ [30], i.e. the correlation between the frequency of transitions in students' learning processes on LMS, to detect details on students' behavioural patterns based on learning event log data.

Details of Random Matrix Theory and its application in this research will be further introduced and discussed in Section 4.3.

2.4 Application types in EDA/LA

In terms of their application, approaches using EDM and LA subdivide into four major categories: computer-supported behavioural analytics (CSBA), computer-supported predictive analytics (CSPA), computer-supported learning analytics (CSLA) and computer-supported visualisation analytics (CSVA) [8]. Such types of applications have also been investigated in previous reviews [148, 14, 131]. With respect to the scope of this research, we mainly focus on the first two categories, i.e. CSBA and CSPA. While CSBA refers to the efforts of discovering insights and patterns in students' learning behaviours, CSPA applications aim to predict students' academic performance based on learning attributes such as grades and participation.

2.4.1 Computer-supported Behavioral Analytics

Computer-supported Behavioral Analytics refer to EDM/LA applications which can uncover significant patterns and reveal valuable insights into students' learning behaviours [85]. Most EDM/LA applications in CSBA focus on using real-time data to monitor the progress of learning new knowledge [8]. Usage behaviours and engagements can be found based on user interactions on an LMS system, to provide educators with deep insights regarding learners [54]. For example, one may detect less common student behaviours by considering the connection between the students' online activities and their final grades [152]. It has been also reported that

log data, which reflect learners' interactions with the LMS, can be utilised to detect successful learners using data mining techniques [108].

The relationship between the learning behaviours and performance of students was investigated by other researchers [39, 38]. A very basic example of this [91] uses a small number size of students. The authors set up a web programming class with 13 participants, using the 'web-based programming assisted system for cooperation'. Although this pilot research merely used data from an experimental class, some evidence for the relationship between learning behaviour style (e.g. completely independent, imitating, self-improving using assistance) and learning outcomes in programming education has been detected and, as such, the topic is worth further investigation. The effect of the diversity of learning behaviour styles on learning scores and satisfaction has also been tested, using data from an online forum and survey data from 144 students [162]. While these efforts consider a wide range of learning activities, they have been carried out with small sample sizes so survey data was still required for the analysis.

Regarding particular learning activities in programming education, Practice has been shown to be essential for improving students' programming skills and students should be given opportunities to practice and receive constructive feedback [21]. In [26], the authors have developed metrics for use as formative assessment tools to analyse (exam passed and failed) students' learning patterns. These approaches have focused mostly on practical activities such as coding and solving programming tasks. Such research can be improved by considering more learning activities in programming study progress along with coding, e.g. reading lecture notes and labsheets. For example, the authors in [40] also included time students spent on lecture slides along with lab programming activities to identify "at-risk" students in an Assembly Language Programming class.

The COVID19 pandemic has been found to have a significant impact on higher education students in terms of their learning behaviours and satisfaction [11, 9]. There have been a number of studies on students' learning behaviours under the circumstances of the COVID19. It has been remarked, for example, that the pandemic

has transformed teaching and learning approaches [100]. While studying from home, students may have more time to get involved in new activities (e.g. writing poetry, doing exercises or exploring third-party training materials). Therefore, they may require a certain degree of self-regulation skills to manage their learning process [6]. The relationship between the viewing duration of lecture videos and learner completion rates in teaching and learning has also been observed [187]. While the study was carried out on students in a prosthodontic programme, the results are interesting nonetheless given the prevailing pandemic circumstances. The authors found that students typically demonstrate a negative perception of online learning [144], causing psychological distress [83]. In the context of autonomous learning, studying in the COVID19 lockdown has been found to change students' learning strategies to a more continuous habit, rather than merely studying on certain weekdays [77]. To broaden the analysis of the change in learning behaviour due to the pandemic effect, in this thesis, we investigate the usage of course resources, delivered throughout the year in the context of programming education.

2.4.2 Computer-supported Predictive Analytics

Prediction of student performance has been one of the most popular topics in Learning Analytics in recent years [145, 115]. Data mining techniques have been shown to have potential to help discover knowledge and hidden patterns underlying a large volume of data and make predictions for learning outcomes [103]. In these works, the authors conclude that EDM and LA predictive applications can contribute to the enhancement of the current learning and teaching experiences. Three primary objectives that have been shown to be significant in the literature are evaluation of course material items, monitoring students' learning and dropout detection.

Monitoring students' learning progress

One of the essential aspects in higher education is evaluation and monitoring students' learning progress [57] which is expected to provide valuable insights that help

educators and learners to make decisions in higher institutions. Data mining tools and techniques can be used to uncover hidden information and detect unexpected behaviours from students' learning data [125]. In addition to the above, the relationship between cognitive skills and reading acquisition has been verified by using data mining [127]. A common application of monitoring students' learning progress using EDM/LA is to predict, as early as possible, the possibilities of student failure before the course ends [22]. In [150], the authors improved the prediction of students' final grades based on the level of engagement of students in online discussion forums. A similar application has been implemented in [184], where the author utilised students' activity log data collected from a computer-supported collaborative learning environment to predict students' learning performance. In [154], using K-means algorithm, the authors identified clusters of students based on their learning behaviours and performance in answering questions in science subjects. The learning characteristics of the clusters were then investigated, supporting the process of determining strategies to strengthen students' competence in science.

Evaluation of Course Material using Data Mining Approaches

Evaluating and enhancing learning materials can also be supported by educational data mining applications. EDM/LA techniques can identify the main characteristics that may affect students' learning outcomes based on the data about their usage and interaction with the learning material items in the courses [35]. Based on the analysis of such data, coursework, study plans and classes schedules can be constructed [148] so that they can best fit the knowledge and current ability of learners. For example, the understanding of how students react to the course materials can help to adjust the complexity of learning exercises and lecture notes [129], as well as provide supportive feedback to the learners [142], which ultimately can lead to optimise overall students' learning performance [112].

Detection of students' dropout possibilities

To deal with the issue of students dropping out from higher education courses, researchers have used many approaches to uncover factors that inhibit college students' overall performance at various academic levels. Despite such efforts, there has been no agreement in terms of the best way to understand the drop-out intention of students regarding pedagogical style or course activities. For example, in [57], the authors predicted the dropout rate of students in two academic institutions after the first semester of their studies, using students' learning attributes (e.g. historical grades, personal details) as tabular data derived from the local LMS. This research found that early failure predictions can be useful for identifying "at-risk" students and making interventions. The learning attributes data can further be enriched, for use in dropout predictions, by taking into account students' social behaviour (e.g. communication via emails) for dropout predictions [18]. Another way to detect possible dropout is by analysing critical factors influencing students' academic performance [134]. Additionally, the authors of [37] use a set of pedagogical actions to estimate students' dropout status, offering the potential of such actions to be used as inputs in such studies (i.e. detecting dropout students).

2.5 Summary

In this Chapter, we have briefly reviewed the key aspects in EDM/LA, consisting of the data, the analysis method and some EDM/LA applications which focus on learning behaviours. In terms of educational data, we have introduced the main data storage formats (e.g. tabular, temporal data) and feature types (i.e. static and dynamic data) that can be collected from learning management systems. Additionally, commonly-used analysis and prediction techniques in EDM/LA have been reviewed, along with their applications in behavioural and predictive analytics.

In this thesis, we aim to address some existing gaps identified from the literature discussed above. First, we extend the research in the EDM/LA domain, i.e., investi-

gating the relationship between student learning behaviours and learning outcomes in programming education, using a large volume of learning log data automatically collected during the study from our bespoke online learning platform over three academic years. The datasets have been recorded from real university programming classes. During the data collection phase, students did not need to follow any additional tasks other than ordinary learning activities in the courses. It is expected that this can help to avoid any sources of experimental setup bias in the collected datasets. The data contain not only student testing results, but the learning behaviours have also been tracked in the form of a temporal dataset, i.e. learning activity logs that store students' interactions with the online learning system. This research also adopts the notion of the "event-log" [174, 151] in EPM as a storage format of the collected dataset.

Regarding data pre-processing, this research deals with the issues of *noise* and *trend* in educational data as stated in Section 1.3. One of the most common methods to pre-process data is Principal Component Analysis (PCA) which has been applied in different areas such as education [186], medical [69] and network security [25]. Although PCA supports the selection of the most relevant features, which may help to unintentionally eliminate noise in the data, the trend effect remains. To the best of our knowledge, none of the existing research directly deals with the issues of *noise* and *trend* in educational data, which, we feel, may have a negative-biased influence on prediction models. Our approach, based on Random Matrix Theory, aims to identify and separate the key information part from the noise. We expect to enhance the performance of the prediction models with cleaned datasets in comparison with original and PCA-based processed datasets. Our work is also one of the first attempts to apply the concept of Random Matrix Theory in order to deal with the problems of noise and trend in the data, which are expected to improve the quality of analysis and prediction.

In terms of analysis methods, instead of using process discovery techniques, which may end up with 'spaghetti-like' and complex process models, we implement Community Detection using extracted behavioural features from the logs i.e. construct-

ing a network structure based on students' learning behavioural data to produce more logical and coherent communities in terms of their learning performance, as well as to generate better predictive models of learning outcomes. The network structure of undergraduate courses and their contributions to students' learning pathways have been investigated using Community Detection approach and Minimum Spanning Tree [68, 141], which are similar to the approach of this research. However, the authors of both studies merely considered the courses' grades from a relatively small number of students. We would argue that more aspects of student learning, e.g., student learning behaviours, can be included to deliver more insightful results. Regarding prediction of students' learning outcomes using learning behavioural features, the authors of [60] also note that from a survey for literature, the evaluation of the *predictability* of the behavioural data at early stages also remains limited.

This study also combines learning behavioural data and other attributes related to students' academic progress into a single manner for prediction. By merging the process data, i.e. event logs of students, and other learning profile data, the prediction of students learning output can be implemented with the support of a range of machine learning approaches. It will provide recommendations on students' learning behaviours. Detail on the methods will be described in Chapter 4.

Chapter 3

Context of the study and Dataset

3.1 Context of the study

This research has been carried out based on four datasets representing the learning behaviour of students and their performance in two programming-related courses in the Computer Science (CS) domain in a Medium-sized Metropolitan University in Ireland. The first course is a first-year introductory programming module delivered to students in the Software Engineering programme. These students generally have the aim of targeting programming-related jobs such as software development. The second course is a programming module taken by first-year Business Computing students who are usually looking for less technical positions (e.g. noncoding roles) in an IT-related field. We denote the two courses as Course#1 and Course#2, respectively.

In both courses, learning material items are provided to the students on a weekly basis. Course items include general course information, lecture notes, labsheets, and programming tasks. Students are expected to read the lecture notes during a lecturing session. In a lab session, students should follow instructions and examples in labsheets and do given programming tasks. The students' solutions to the tasks are uploaded and tested automatically by the system. The course items are delivered in the form of web pages on the bespoke online learning system.

There are three formative assessments in Course#1, which take place in weeks 4, 8 and 12 where 12 is the final week of the semester, while Course#2 students have to take two assessments in the form of two lab exams in weeks 6 and 12. All lab exams are mandatory and carry the same weight in the overall assessment mark. Students are therefore required to take the exams seriously by doing all given programming tasks as much as they can. Students are likely to find the last assessment in each module to be the most challenging, requiring a comprehensive understanding of the course knowledge to solve the given problems and also carrying the most marks. Therefore, the results of the last exam will be used as a basis for further analysis in this research.

The two courses are expected to provide students with fundamental knowledge and skills in Python programming. Since they are prerequisites for their specific programmes, both modules are mandatory and key to their overall programme outcome goals. The motivation for students to take both courses is the same, as they cannot follow the curriculum of the programme without deep understanding of these modules. Hence, we assume that students are likely to take these modules seriously and fully participate in learning activities to maximise their learning benefits, although Software Engineering students might be expected to pay more attention on programming modules than Business Computing students.

The differences between the two modules is related to the level of knowledge and the requirement of the tasks, e.g. the tasks of Course#1 may require students to apply more advanced algorithms to solve the given problems. In fact, in Course#1, students are taught more advanced concepts in programming and given more challenging exercises, compared to students in Course#2. As a result, students in Course#1 generally have more activities in learning than Course#2 students. In other words, while Course#1 can be seen as a typical programming course for Software Engineer students, Course#2 represents a programming course for learners whose objectives may be toward business but still need programming skills at a certain level. It is important to note that both courses have the same coordinator and there were no major changes in the curriculum over the two academic years.

Table 3.1: Datasets information.

Dataset	Number of students	Number of events	Average events per student
Course#1-2018	112	1,054,394	9,414
Course#1-2019	151	1,484,297	9,829
Course#1-2020	128	1,589,216	12,415
Course#2-2018	62	211,855	3,417
Course#2-2019	48	200,006	4,166
Course#2-2020	65	216,148	3,325

Therefore, the learning motivation of students is not expected to significantly vary. However, their behaviours can be distinct due to the differences in the level of course requirements. As a result, these datasets can reflect the diversity of learning characteristics of students' learning behaviours, giving a good quality of data.

The collection and usage of the data has been approved by the Research Ethics Committee of Dublin City University (REC Reference number DCUREC/2019/156).

3.2 Dataset and Learning Event logs

We formalise the course material items in this context as material type (i.e., *General, Lecture, Labsheet and Practice*) combined with the corresponding week, for example *Labsheet_1* means the labsheet used in week 1. For the general documents, e.g. course information and technical instructions notes, we denote them as *General*. Students' interactions with the items (e.g. mouse clicking or scrolling, highlight a piece of text or switching between two items) are logged automatically into the database. Brief information about the collected data can be seen in Table 3.1.

We can consider a real-life scenario of students learning programming on the online learning system as follow: On a day in Week 5, student $s1$ read a labsheet for a task instruction. While reading a labsheet, the student also switched between lecture notes and the labsheet two times, and another two mouse events on the lecture page were logged; The student could then write the code to solve a given task and upload it to the system via the submission portal. All these learning events

Table 3.2: Example of event log of student $s1$ on two days in week 5 in Course#1 module.

Event Item	Timestamps	Student id	
1	Labsheet 5	2018-08-12 14:30:00	s1
1	Labsheet 5	2018-08-12 14:35:00	s1
1	Lecture 5	2018-08-12 14:36:00	s1
1	Labsheet 5	2018-08-12 14:45:00	s1
1	Lecture 5	2018-08-12 14:49:00	s1
1	Labsheet 5	2018-08-12 14:50:00	s1
2	Practice 5	2018-08-13 11:59:00	s1
2	...	...	s1

of the student $s1$ were recorded and stored as the event data structure called the *event log* which can be seen in Table 3.2 as an example.

We adopt the format of *event log* in *Process Mining* [174] to store the students' learning behaviour. An event log includes a collection of events implemented in chronological order. Each event belongs to a learning trace which refers to the sequence of events of a student within session. Event logs may contain other attributes such as timestamps, participants, and results. In the context of this research, a student's learning event log comprises the following information:

- *Trace id*: A trace refers to a sequence of learning events of a student over a day (i.e., within 24 hours). For example, Table 3.2 illustrates two learning traces associated with 12 August and 13 August 2018 of the student $s1$.
- *Event Item*: An event item refers to an item of course material of the corresponding week where students' interaction with the system are logged. For example, the first row in Table 2 indicates that there was an event on the Labsheet_5 course material item generated by student s1 at 14.30 on 12 August 2018.
- *Timestamp*: Timestamps refer to the date and time when the corresponding event occurred recorded by the system. The timestamp is essential information, as it will be used for ordering events and reflecting the behaviour of students.

- *Student id*: This refers to the unique identity of students. Please note that in practice, we merge many students' event logs into a single event log dataset (or combined event log) which is convenient for further analysis. Hence, the student id field is important for further calculation later on.

3.3 Exploratory Analysis

This section explores some general characteristics of the collected datasets for Course#1 and Course#2 over the three academic years (2018, 2019 and 2020) mentioned above. For each course's dataset, we describe how students interacted with the University's bespoke learning systems, i.e., the total number of learning events made by students over the study period in the semester, separated by weekdays. We also investigate the students' interactions with different types of learning item in the courses. The difference behaviours between the higher and lower performing cohorts of students are also highlighted.

3.3.1 Course#1 datasets

Figure 3.1, 3.2 and 3.3 illustrate the total number of activities in each week day recorded from students in Course#1 over the three academic years of 2018, 2019, and 2020, respectively. The data for Course#1-2020 represents students' learning activities during the lockdown of COVID19 when the University's courses were delivered completely online and students were not allowed to physically go to college. Although it was a remote setting, students were still expected to attend lectures and lab sessions synchronously on certain days in a week.

In all figures of Course#1 datasets, one can notice the days in a week where lecture and lab sessions were scheduled, when showing the most significant number of activities. In 2018 and 2019, students had a similar schedule with lectures on Thursday and lab sessions on Tuesday while in 2020 when during the lockdown, the schedule was slightly changed, i.e., lectures on Thursday but lab sessions on

Wednesday. In addition, the length of the Course#1-2020 was 10 weeks only due to a change in timetable format, instead of 12 weeks as usual in the two former academic years. On the other hand, on the days of the week when there were no lectures or lab sessions, a significantly smaller number of activities can be observed. These figures indicate that students generally did not use the learning system beyond the scheduled learning time. We can also notice the week in the semester where students needed to take a lab exam based on the unusually high number of activities on a non-lecturing weekday. In 2018 and 2019, students had lab exams on Wednesday of week 4, 8 and 12, while in 2020, lab exams were conducted on Tuesday of weeks 4, 7, and 10. It can be said that, other than on the lecturing and practice day, most of the students were only active on the pre-assessment days.

Furthermore, the number of students' activities also decreased from the beginning to the end of the semester in all courses. This observation may imply that students appear to be less focused over time during the study period. They might be absent from the class and skip doing given programming exercises, and hence did not interact with the learning system.

The usage of course material items (i.e., Lecture, Practice, Labsheet and General items) in Course#1 by higher and lower performing students can been seen in the boxplots in Figures 3.4, 3.5 and 3.6 below. In this research, the higher and lower performing cohorts of students in a course are defined based on students' performance in the final lab assessment of the course. On finishing a programming lab examination task, students submit their codes to the system and receive the results as "correct" or "incorrect" submission. A submission is considered "correct" if it passes all test cases which are predefined by the instructor. Each task is given the same proportion of marks, and the overall mark is given to students after the exam is finished. A student whose grade is less than 40 out of 100 is labelled as "lower performing', otherwise, that student is considered as "higher performing". In this research, the overall marks of the students have been used for behavioural analysis while labelling is used as a target variable, i.e., "higher performing = 1" and "lower performing = 0" for the evaluation of the predictability of the behavioural data for

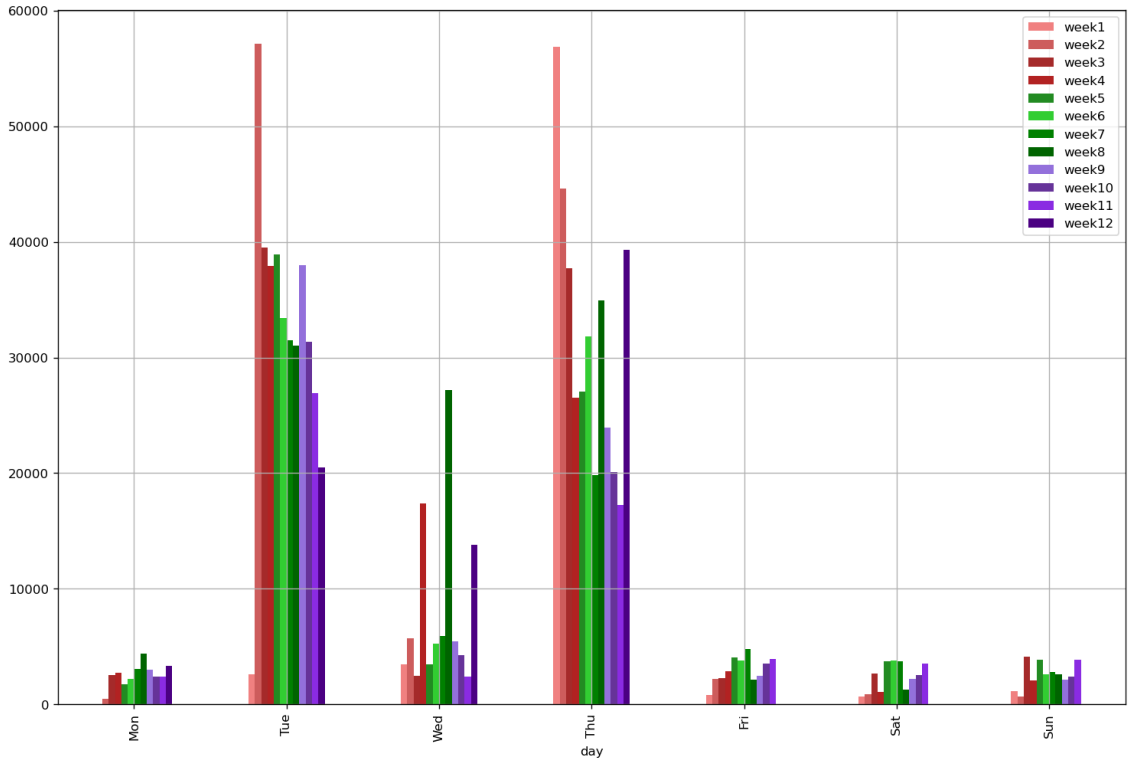


Figure 3.1: All student activities in Course#1-2018 (pre-COVID19)

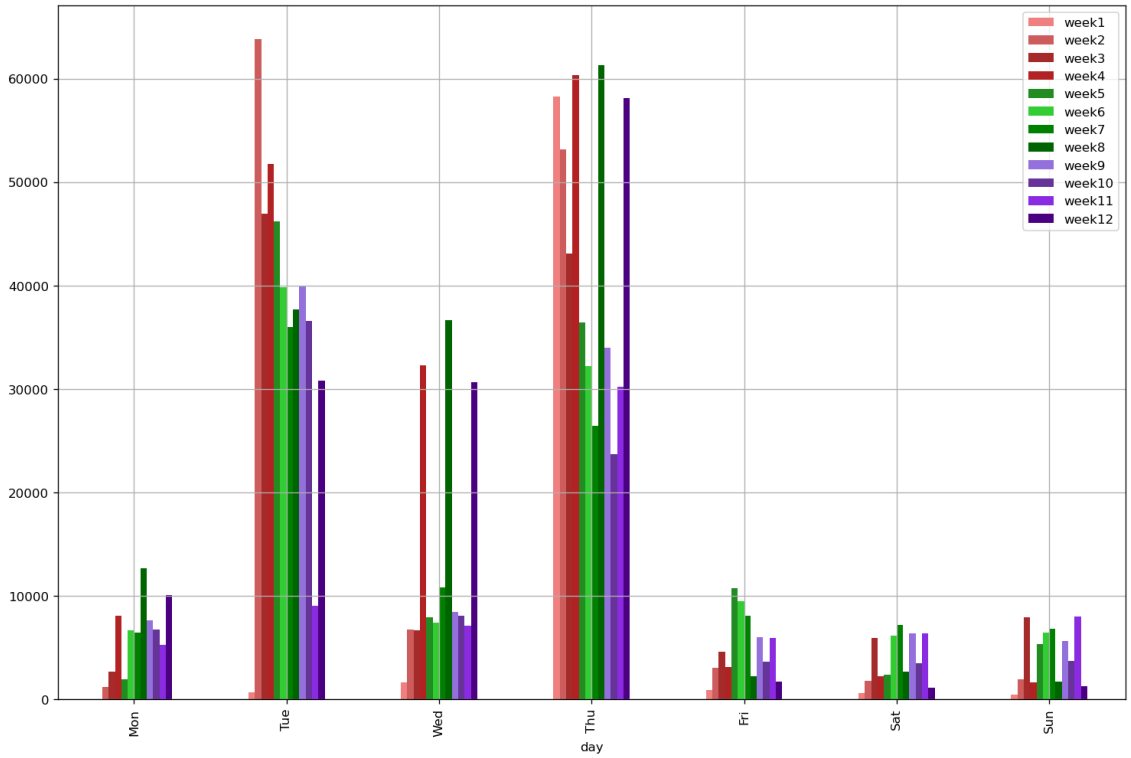


Figure 3.2: All student activities in Course#1-2019 (pre-COVID19)

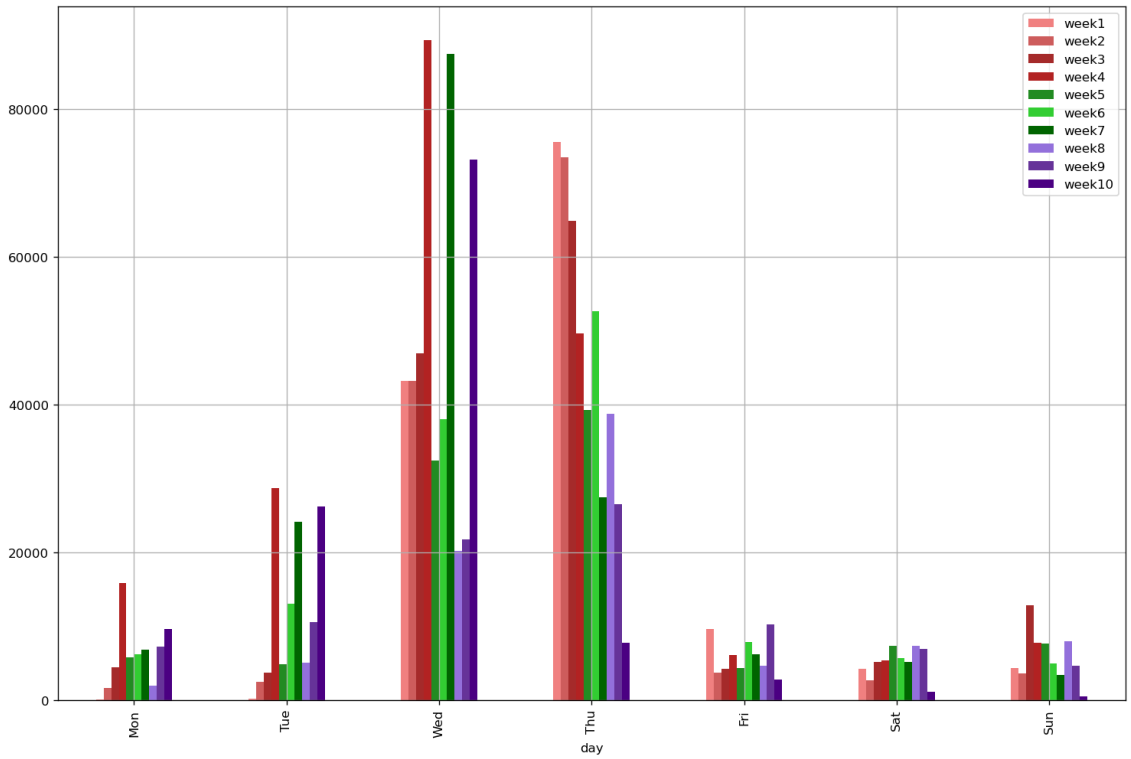


Figure 3.3: All student activities in Course#1-2020 (during COVID19)

the students' learning outcomes.

In general, over the three years of Course#1 recorded, students had the most number of interactions on Practice and Labsheet items. This is an expected result due to the large volume of exercises given in such programming courses in a third-level institution, and students had to read the requirements and instructions in labsheet items for practise. Furthermore, the higher performing students have shown more interest than the rest of the class in interacting with Practice and Labsheet items in Course#1 over the three academic years.

On the other hand, Lecture items are also an important learning material that students had to go through during, and possibly after, the lecturing sessions. Students might also need to read lecture notes while doing programming tasks in practical sessions. Interestingly, in contrast to Practice and Labsheet items, lower performing students appear to be more active in Lecture items. In both academic years before COVID19, lower performing students were likely to carry out more learning

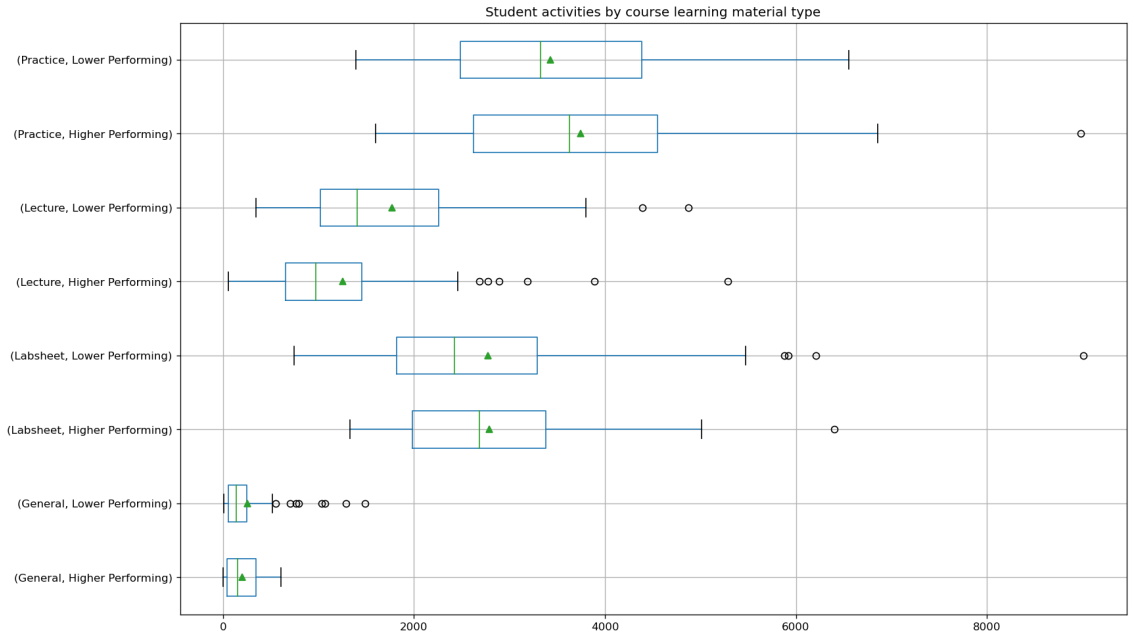


Figure 3.4: Students' learning activities by learning material item types in Course#1-2018 (pre-COVID19)

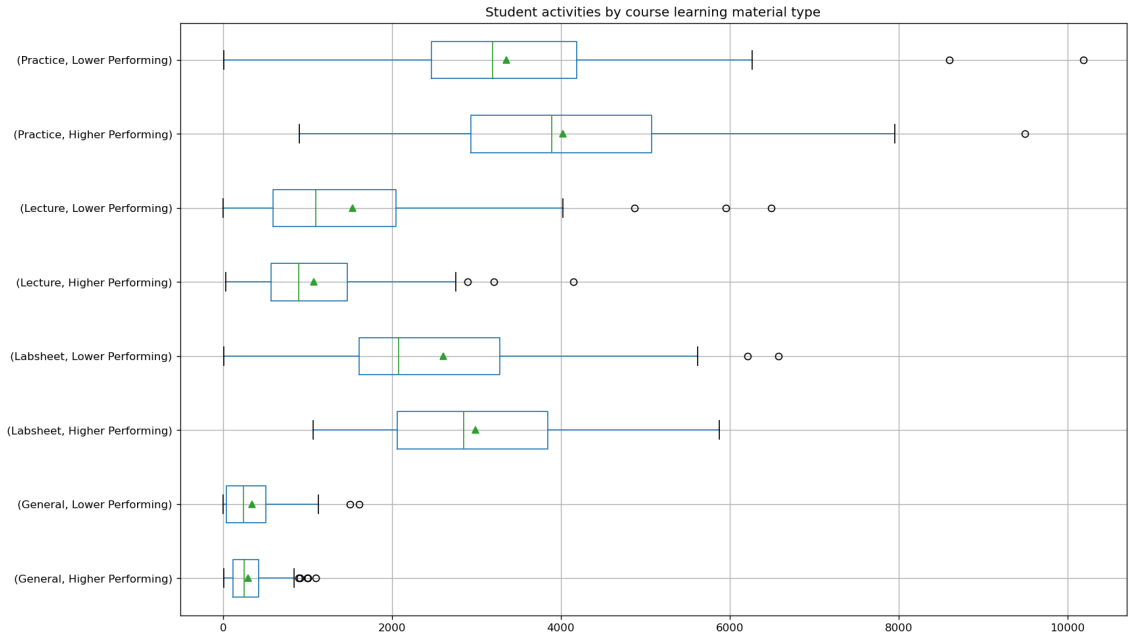


Figure 3.5: Students' learning activities by learning material item types in Course#1-2019 (pre-COVID19)

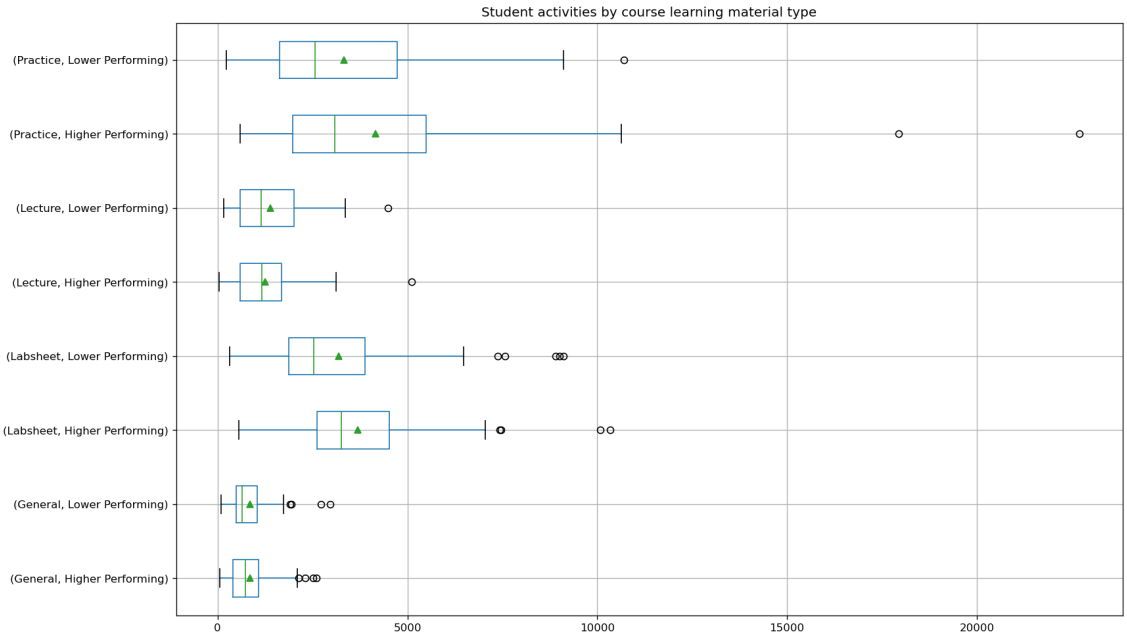


Figure 3.6: Students' learning activities by learning material item types in Course#1-2020 (during COVID19)

activities on Lecture items in comparison with higher performing students (Figures 3.4 and 3.5). However, the difference is not significant in Course#1-2020 which was during the pandemic (Figure 3.6). It is possible that before COVID19, students could physically attend the class where they had a chance to directly interact with the lecturer, e.g., students could see the examples on screen and ask questions immediately. As a result, higher performing students could understand the new knowledge and may not have needed to spend much time on lecture note items after the class while the lower performing cohort had to re-visit the lessons and therefore had more activities in Lecture items. In contrast, during COVID19, studying online might have restricted communication between students and lecturers and neither student cohorts had many options other than to rely on the provided learning materials.

Meanwhile, students had shown the least attention to General items, as shown by the absence of much difference between the higher and lower performing cohorts of students in the interaction with these items. This is understandable because general items merely contain course information, such as the timetable and exam schedule,

and therefore may not have much contribution in students' learning success, although students still need to follow those items to ensure that they will perform all required tasks and lessons correctly.

3.3.2 Course#2 datasets

As with sub-section 3.3.1, this sub section provides a birds'-eye view of the Course#2-2018, 2019 and 2020 datasets. Figures 3.7 and 3.8 show the data for all student activities in Course#2-2018 and Course#2-2019 that occurred before the COVID19 pandemic. Based on the figures, it can be seen that both roll-outs of the course have the same learning schedule and activities. In general, students in both courses were mainly active on Thursdays, which were scheduled for both lecture and lab practice sessions. In addition, students had more logged learning activities on Thursday of the first half weeks of the semester in comparison with the ending weeks.

On the other hand, there are only a small number of learning activities recorded on the other days of the studying weeks. This phenomenon might indicate that most students in Course#2 only studied, i.e., in terms of activity on the learning system, on lectures and lab days. However, an exception can be witnessed on Wednesday of Week 6 and 12 in both Figures 3.7 and 3.8, when students had a lab exam so that they needed to complete the exam tasks and submit the solutions to the system.

In comparison with the above, Figure 3.9 illustrates the students' learning activities in Course#2 in the 2020 academic year when a lockdown was imposed due to the effect of the COVID19 pandemic. The course was delivered completely online and the students mainly interacted with the lecturer and other students through online meeting tools. The lab exams were also conducted at home instead of in the laboratory rooms at the University. Overall, we can observe the similar learning behaviours with that of students in the previous years. In particular, students were most active on lecturing or practise days, i.e., on Friday and Tuesday in Course #2-2020. Students had also been being less active over the semester and only had more interactions with the learning system for the preparation of the exams.

With respect to the comparison between Course#2 and Course#1, it seems that Software Engineering students in Course#1 were more active in checking learning material items during weekend (see Figures 3.1, 3.2 and 3.3) while Course#2 students, who were studying in the Business Informatics domain, had almost no activity on these days. This may reflect the fact that Course#1 is more challenging than Course#2 in terms of the level of knowledge delivered and difficulties of exercises, requiring the students to spend more time on studying. It may also be due to the fact that the Software Engineering students see this as a really important course and it is also worth double the number of credits.

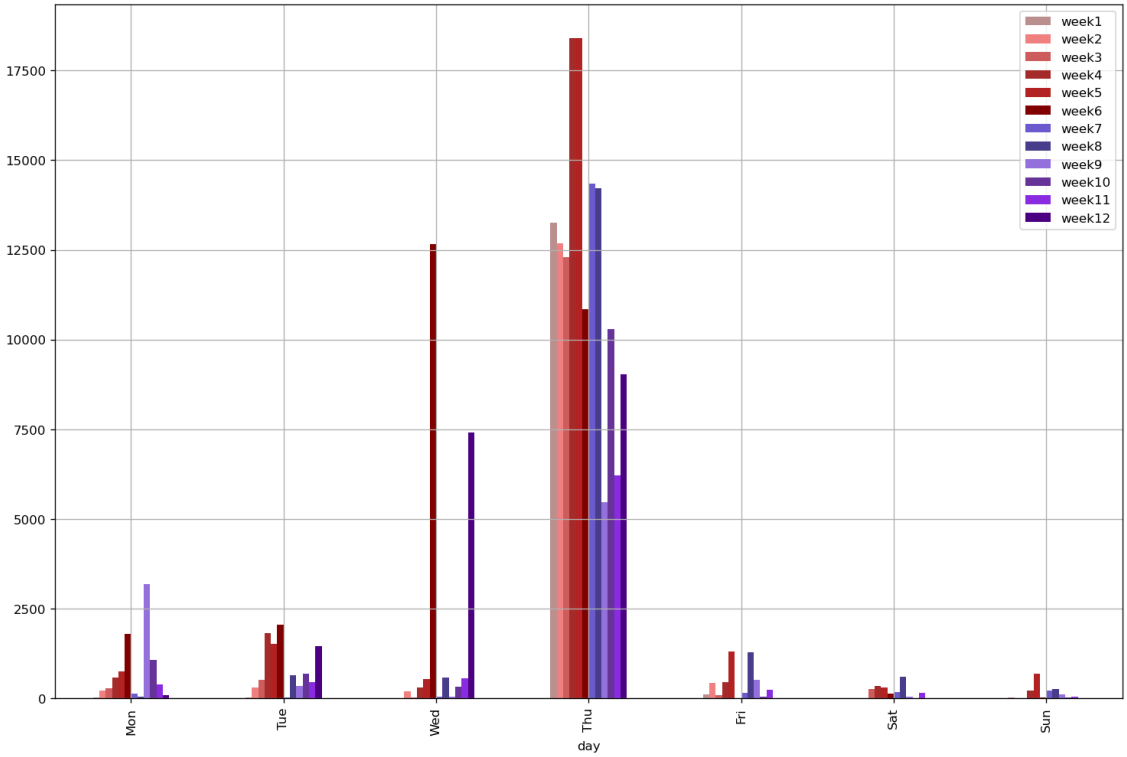


Figure 3.7: All student activities in Course#2-2018

Regarding the difference in learning behaviours between the two cohorts (i.e., higher and lower performing), it can be seen in Figures B.6, 3.11 and 3.12 that the data for Course#2 are roughly similar to that for Course#1. In particular, students in Course#2 were also active the most in practical-related items. Higher performing students in Course#2 also appeared to have more learning activities in practical-

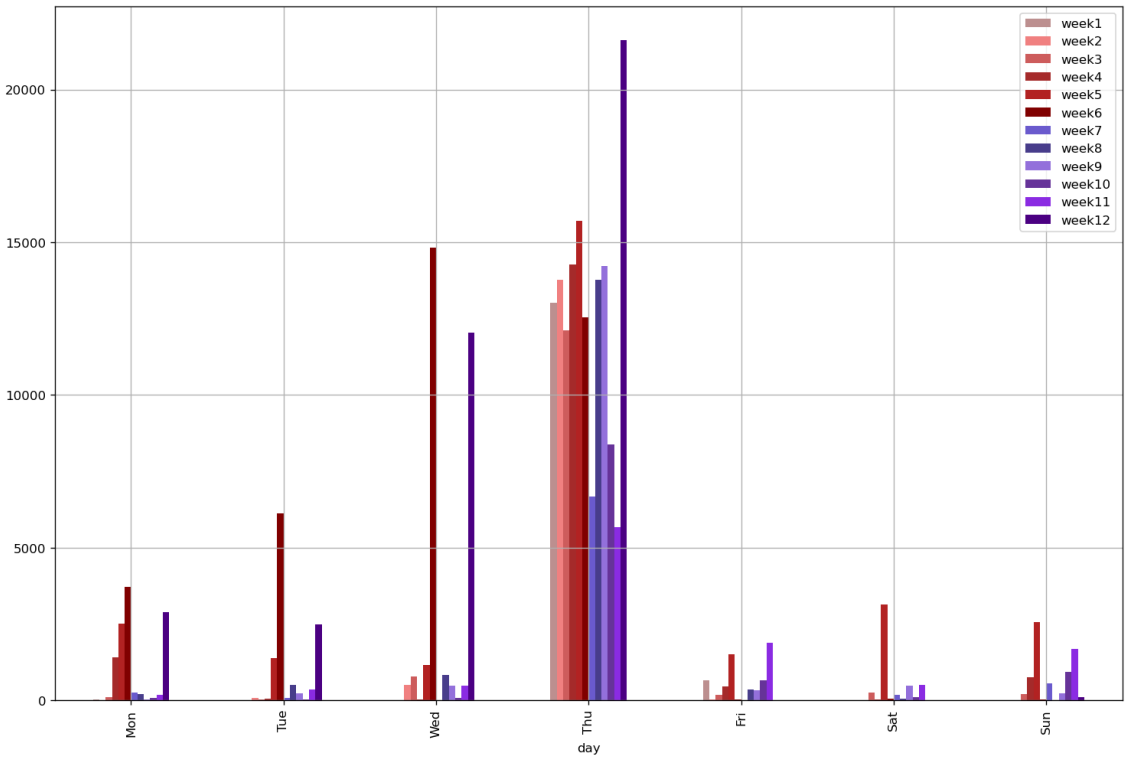


Figure 3.8: All student activities in Course#2-2019

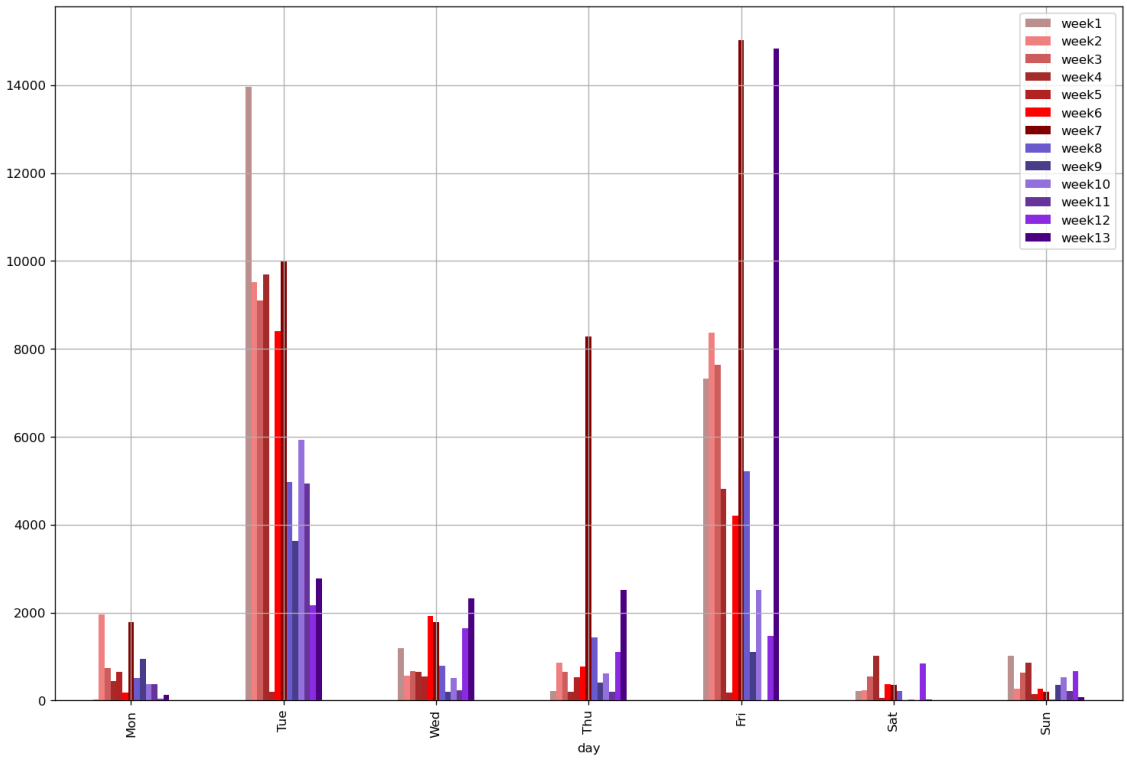


Figure 3.9: All student activities in Course#2-2020

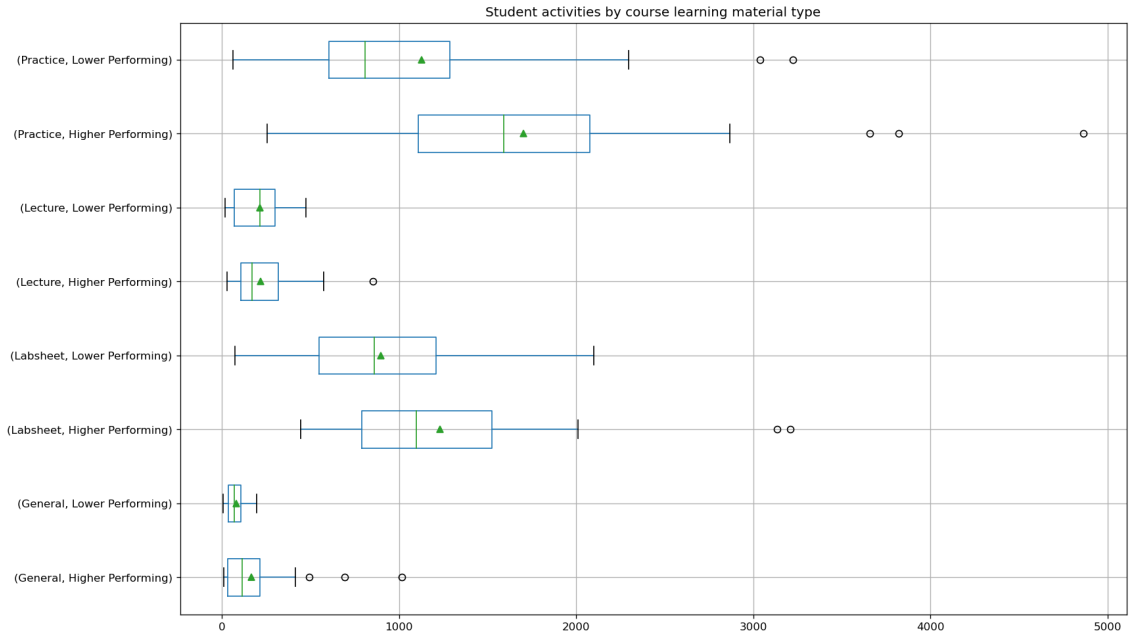


Figure 3.10: Students' learning activities by learning material item types in Course#2-2018

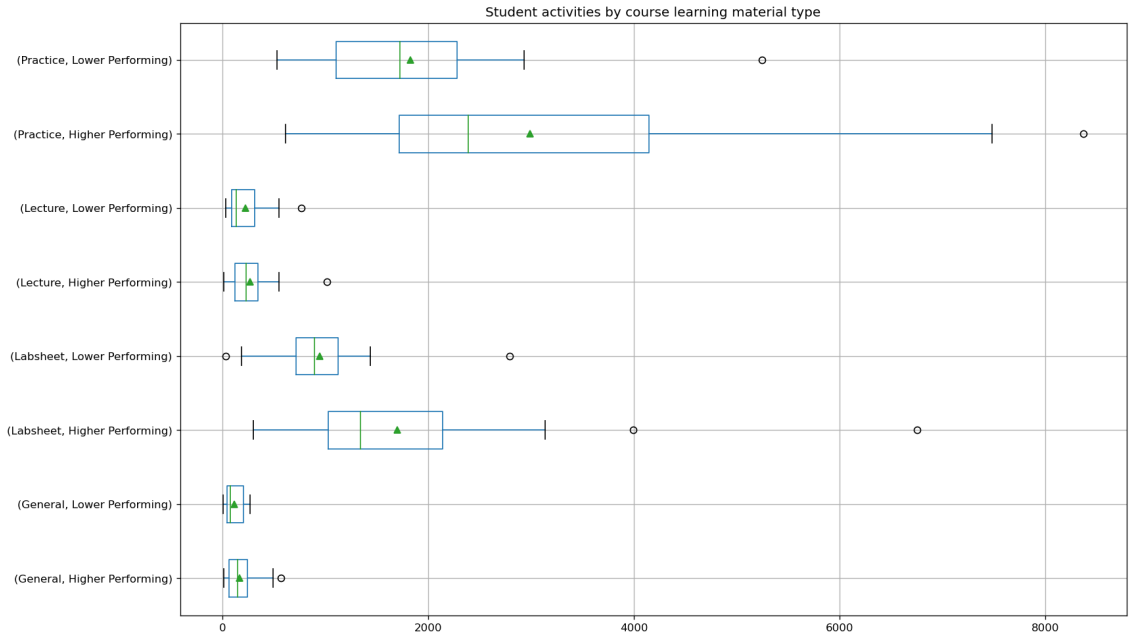


Figure 3.11: Students' learning activities by learning material item types in Course#2-2019

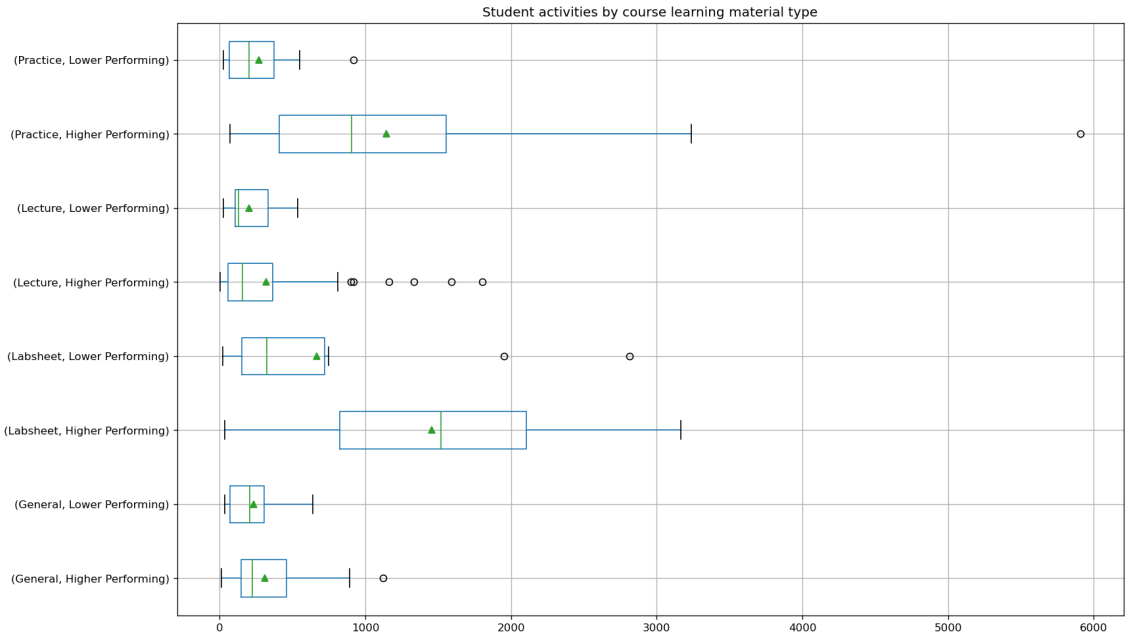


Figure 3.12: Students' learning activities by learning material item types in Course#2-2020

related items (Practice and Labsheet) than lower performing students. On the other hand, the difference in the number of interactions in Lecture items between the two cohorts of students is not significant during the three academic years of Course#2. In other words, it is hard to detect any deviation in the level of reading lecture notes between the higher and lower performing cohorts in Course#2 based on such an exploratory analysis.

3.4 Conclusion

The exploratory analysis of the collected datasets reveals several large patterns in students' learning activities. Students have been shown to be gradually less active during the semester. They were also only active on the days when lecturing and practice sessions were took place and gradually paid less attention on other days of the week.

In terms of the difference in learning behaviours between cohorts of students, we notice that higher performing students are likely to be more active than lower

performing students in doing programming tasks, represented by a higher number of interactions with practical-related items (i.e. Practice and Labsheet) in all courses over the years. Regarding Lecture items, there were also no noticeable gaps in reading lecture note items between the two cohorts of students in Course#2 while learning behaviours of student cohorts in Course#1 are different. Before the COVID19 pandemic, lower performing students in Course#1 appear to have had more interactions with Lecture items than higher performing students in Course#1. However, this difference was not significant during the COVID19 confinement.

Generally, it can be said that the data exploration process has revealed the relationship between learning patterns and learning outcomes of students. Results from previous research efforts have also concluded that learning behaviours are correlated with learning performance [158, 39]. Students *at risk* in study progress will show signs of this risk in their learning behaviours [116]. Learning behaviours typically refer to participation frequencies in learning activities such as login frequency [89] or mouse clicks [66]. Yet, the role of course material items or learning documents has not been commonly investigated. For example, it is possible that students with a difficulty in learning programming can be more active in reading lecture notes because they do not fully understand the abstract concepts. In this case, although a student has a high number of times logging in and interacting with the system, it is not guaranteed that the student will achieve a high result in the exam due to the difficulty in understanding.

However, there are several problems noticeable in the data due to the nature of educational contexts, which require further data processing and analysis to verify the identified learning patterns. The Figures 3.4, 3.5, 3.6, B.6, 3.11, and 3.12 generally do not show a great difference between higher and lower performing students in the learning activities with various learning items. This means that there are still a number of students in this cohort (e.g., lower performing), who have shown learning behaviours that are similar to the characteristics of the other cohort (e.g. higher performing). For example, there were also students who completed many practical exercises, but still failed the exam. We argue that the effects of noise and trend in the

collected data contributed to this complexity of the analysis. In other words, simple statistical methods cannot fully explain the learning patterns, and the analysis would need to be enriched with more thorough methods.

In the next Chapter, we propose our approach to extract relevant features of learning behaviours with regards to the use of course material items as well as showcase how the noise and trend effect could be removed from the extracted data. More sophisticated methods will also be applied, e.g. using Community Detection algorithms to identify better representative communities.

Chapter 4

Learning behavioural features and Analysis methods

4.1 Introduction

In this chapter, we describe the methods used in this research. Generally, our methodology can be seen in Figure 4.1. Firstly, we collect student log data from the two programming courses namely Course#1 and Course#2 over the three academic years, i.e., 2018 (2018/2019), 2019 (2019/2020) and 2020 (2020/2021). The collected data will be stored as learning event logs, followed by the extraction of learning behavioural features (see Section 4.2). Secondly, the extracted datasets will be cleaned using our proposed method which is built based on the assumptions of Random Matrix Theory (see Section 4.3 and 4.4). These methods are expected to contribute to answering *Sub RQ1* in this thesis. Then, after being cleaned, the datasets will be ready for further analysis and prediction. In particular, Community Detection techniques will be used to analyse the relationship between groups of learning patterns and students' academic outputs. The method is described in Section 4.5 and aims to answer *Sub RQ2* and *Sub RQ3*. Additionally, we investigate the predictability of the students' learning outcomes using the learning behavioural datasets based on the approach in Section 4.6, answering *Sub RQ4*.

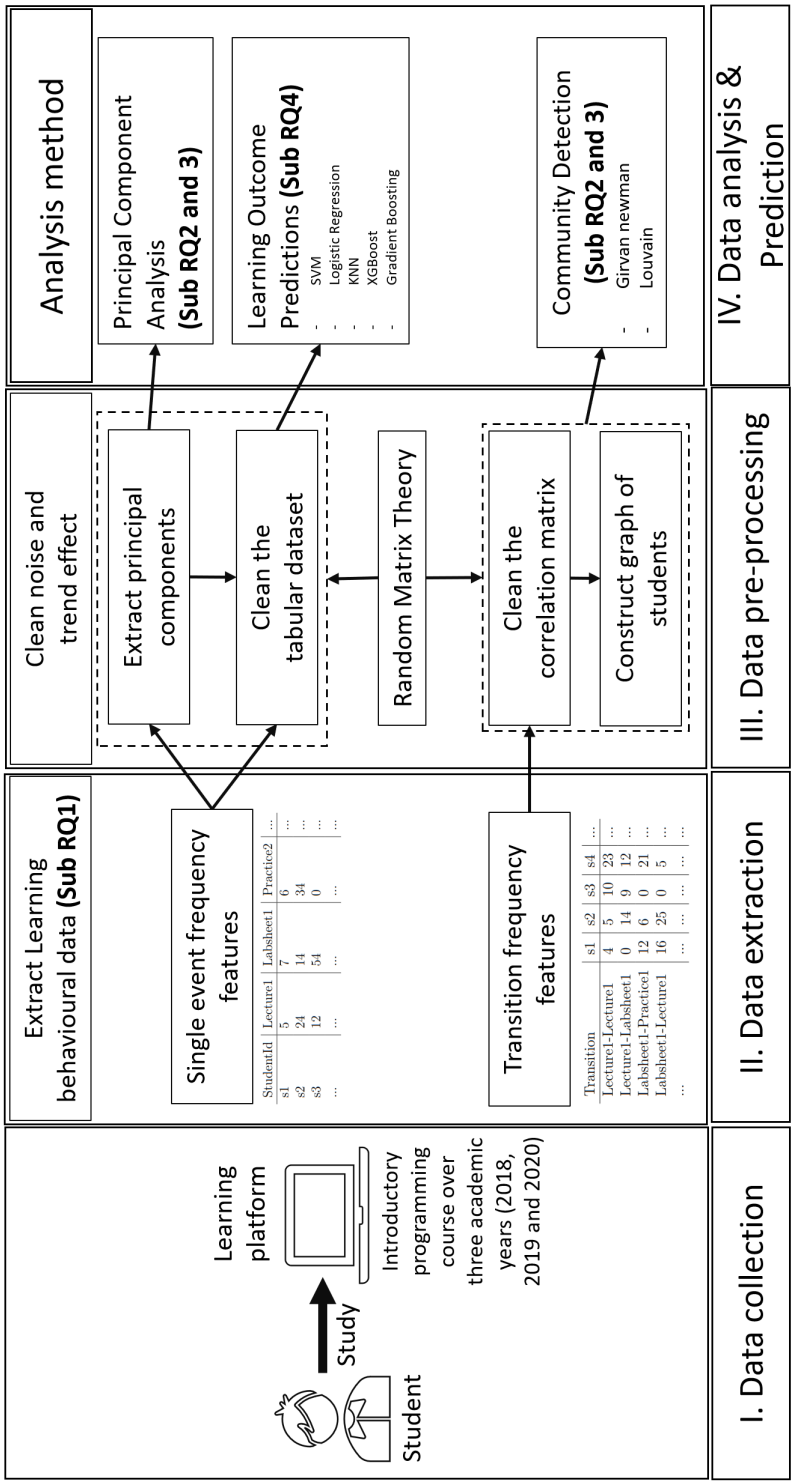


Figure 4.1: Research methods: Steps to be conducted in this thesis

Table 4.1: Example of student-event item data matrix.

StudentId	Lecture1	Labsheet1	Practice2	...
s1	5	7	6	...
s2	24	14	34	...
s3	12	54	0	...
...	...	...	...	...

4.2 Learning event log and behavioural features

4.2.1 Single Event Frequency features

We use two types of features to reflect the learning behaviour of students. The first type is the *single event frequency features*, i.e. the number of events that occurred in each course material item. *Single event frequency features* extracted from the event log can be arranged as a *student-event data matrix* where each column refers to the number of events on a material item generated by students and each row is the data for each student. An example of a *student-event data matrix* can be seen in Table 4.1.

4.2.2 Transition Frequency features

The second type of students' learning behaviour features is *transition frequency features*, i.e., the number of occurrences that a student moves from an event on a course item to another event. Please note that the two events can be on the same item or two different items. We use the term *transition* to denote this phenomenon of moving between consecutive events. The *transition frequency features* can be arranged as *transition-student data matrix* where the rows refer to *transition frequency features* and the columns are the data for the students. An example of a transition data matrix of an event log can be seen in Table 4.2. The value of *Lecture1-Labsheet1* for student $s2$ equals to 14 indicates that student $s2$ performed an event 14 times on *Lecture1* directly before the next event on *Labsheet1*. Please note that if the two materials are the same, e.g., *Lecture1-Lecture1*, the transition reflects a loop in the learning process, i.e. the student keeps working on the same course item *Lecture1*.

Table 4.2: Example of transition-student data matrix.

Transition	s1	s2	s3	s4	...
Lecture1-Lecture1	4	5	10	23	...
Lecture1-Labsheet1	0	14	9	12	...
Labsheet1-Practice1	12	6	0	21	...
Labsheet1-Lecture1	16	25	0	5	...
...	...	...	...	...	...

In this work, the *student-event item data matrix* has been used for the learning outcome predictions and the *transition-student data matrix* has been used for analysing the relationship between students' learning behaviour and their assessment performances.

With this noisy data collected, we pass into description of how it is processed and cleaned to better support our further analysis and prediction. The detailed method to clean the effect of noise and trend in the data is discussed in the next sections.

4.3 Random Matrix Theory and Principal Component Analysis

Given a $m \times n$ data matrix $\mathbf{G}$ extracted from an event log, we can normalise the matrix $\mathbf{G}$ as $\mathbf{G}(n)$ as follows [160]:

$$\mathbf{G}(n)_j = \frac{\mathbf{G}_j - \overline{\mathbf{G}_j}}{\sigma_j} \tag{4.1}$$

where $\mathbf{G}(n)_j$ is the jth column of the matrix $\mathbf{G}(n)$; $\mathbf{G}_j$ is the jth column of the matrix $\mathbf{G}$. In the case where $\mathbf{G}$ is *a transition-student data matrix*, $\mathbf{G}_j$ denotes the frequency of all occurred transitions of a student j. For example, in Table 4.2, $\mathbf{G}_j$ refers to column $s1$, $s2$ etc. On the other hand, if $\mathbf{G}$ is the *student-event item data matrix*, $\mathbf{G}_j$ denote the frequency of learning events associated to the learning item j of all students. For example, in Table 4.1, $\mathbf{G}_j$ refers to column *Lecture1*, *Labsheet1*, *Practice2* etc. $\overline{\mathbf{G}_j}$ is the mean value of $\mathbf{G}_j$ and σ_j is the standard deviation of $\mathbf{G}_j$. In other words, $\mathbf{G}_j$ and $\mathbf{G}(n)_j$ reflect the learning behaviour of the student j.

The correlation matrix $\mathbf{C}$ can be expressed in terms of the inner product of $\mathbf{G}(n)_i$ and $\mathbf{G}(n)_j$ as follows:

$$\mathbf{C}_{ij} = \left\langle \mathbf{G}(n)_i, \mathbf{G}(n)_j \right\rangle \tag{4.2}$$

We note that $\mathbf{C}_{ij} \in [-1; 1]$. It may be noticed that the correlation $\mathbf{C}_{ij}$ can reflect how similarly two students i and j interacted with course material items. If $\mathbf{C}_{ij} > 0$, the transitions of the two students i and j increased together and the students behaved similarly in the course. Conversely, if $\mathbf{C}_{ij} < 0$, the two students tend to behave differently on the learning system.

The eigen-decomposition of $\mathbf{C}$ can be shown (e.g. see [160]) to be given by:

$$\mathbf{C}V = \Lambda V \tag{4.3}$$

where Λ is a diagonal matrix n x n of eigenvalues λ_i and V is a matrix whose columns refer to the corresponding eigenvectors $\mathbf{v_i}$ of $\mathbf{C}$.

Given a random matrix $\mathbf{A}$ where $\mathbf{A}$ is a matrix $m \times n$ with randomly distributed elements with zero mean and unit variance, it has been shown that [181] the properties of $\mathbf{C}$ can be compared to the correlation matrix $\mathbf{R}$ of the random matrix $\mathbf{A}$ as

$$\mathbf{R} = \frac{1}{m}\mathbf{A}\mathbf{A}^T \tag{4.4}$$

where $\mathbf{A}^T$ is the transposed matrix of $\mathbf{A}$.

According to RMT, the statistical properties of such a matrix $\mathbf{R}$ are known [61]. In particular, when the sample size $m \to \infty$ and the number of features $n \to \infty$, provided that Q-factor $= \frac{m}{n} \geq 1$ is fixed, the distribution of eigenvalues λ of the random matrix $\mathbf{R}$ is given by the Marchenko-Pastur probability density function [135]:

$$P_{\mathbf{R}}(\lambda) = \frac{Q}{2\pi\sigma^2}\frac{\sqrt{(\lambda_+ - \lambda)(\lambda - \lambda_-)}}{\lambda} \tag{4.5}$$

where $\lambda_- \leq \lambda \leq \lambda_+$, λ_- and λ_+ are the lower and upper limits, the eigenvalues

of $\mathbf{R}$, respectively, given by:

$$\lambda_{\pm} = \sigma^2\big(1 \pm \sqrt{\frac{1}{Q}}\big)^2 \tag{4.6}$$

where $\sigma = 1$ due to $\mathbf{A}$ having unit variance.

We note that $\lambda_{\pm}$ are the upper / lower limits of the theoretical eigenvalue distribution. Eigenvalues that fall outside of this range are assumed to deviate from the expected values of the Random Matrix Theory [98]. As a result, by comparing this theoretical distribution with the empirical data, we can identify key eigenvalues containing specific information on the data. This characteristic of the RMT supports the need to clean the effect of noise and trend in the data [135].

The Inverse Participation Ratio (IPR) is additionally used to assess the contribution of eigenvector elements to the corresponding principal component where each element is associated with a column in the original dataset. The IPR of the eigenvector U^k is given by

$$\mathrm{IPR}^k = \sum_{l=1}^{n} \big(u_l^k\big)^4 \tag{4.7}$$

where u_l^k is a component of the eigenvector U^k. We focus on the value of $1/\mathrm{IPR}^k$ which implies the number of eigenvector elements significantly contributing to the PCs. Eigenvector elements can be investigated to observe the common trend in students' behaviours as well as the difference in the behaviours between student cohorts. There are two limiting cases for the IPR:

- When the eigenvector has identical components $u_l^k = 1/\sqrt{N}$, then $IPR = 1/N$
- If one component $u_l^k = 1$ and all others are zero, then $IPR = 1$.

The IPR quantifies the reciprocal of the number of eigenvector components that contribute significantly to each principal component/eigenvalue.

In addition, based on PCA theory, the new n variables $\mathbf{x}_i$, forming a new data matrix $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n]$ can be obtained after Principal Component Analysis of

$\mathbf{G}(n)$ as follows:

$$\mathbf{x}_i = \mathbf{v}_{i1}G(n)_1 + \mathbf{v}_{i2}G(n)_2 + ... + \mathbf{v}_{in}G(n)_n = \mathbf{v}_i^T G(n) \tag{4.8}$$

where $1 \leq i \leq n$, x_i refers to the scores and $\mathbf{v}_i$ refers to the loadings (eigenvectors) of the associated principal component (eigenvalue) i. In other words, each principal component has its own eigenvalue and eigenvector.

We can also reconstruct the original normalised data $\mathbf{G}(n)$ from $\mathbf{X}$ as follows:

$$\mathbf{G}(n) = \sum_{i=1}^{n} \mathbf{v}_i \mathbf{x}_i \tag{4.9}$$

4.4 Noise and trend effect cleaning

We have noticed that, in the practical usage of the online learning system, students may interact flexibly with course material items. Although the students can be given the same instructions and learning pathway, they are free to use learning resources in their own way. This phenomenon appears to create noise in the event log data. On the other hand, as all students attended the same lectures, the learning instructions given to them are the same. As a consequence, all students may interact similarly with course material items which were released sequentially and gradually. We can observe this trend effect in Figure 4.2, for correlations matrix of student transitions for Course#1-2018. Most transitions among students are highly correlated. This issue may limit the chance of detecting differences in learning behaviours among groups of students. Therefore, it is necessary to clean the effect of noise and trend in the dataset before doing any analysis [135].

In this thesis, we adopt, from financial references such as [95, 133], the concept of a "Market Component". This corresponds to the largest eigenvalue of a correlation matrix and represents a cross-market effect (e.g., rise or fall) affecting all stocks. Similarly, the trend effect in a classroom can be reflected by the largest eigenvalue of the correlation matrix of students' learning behaviours.

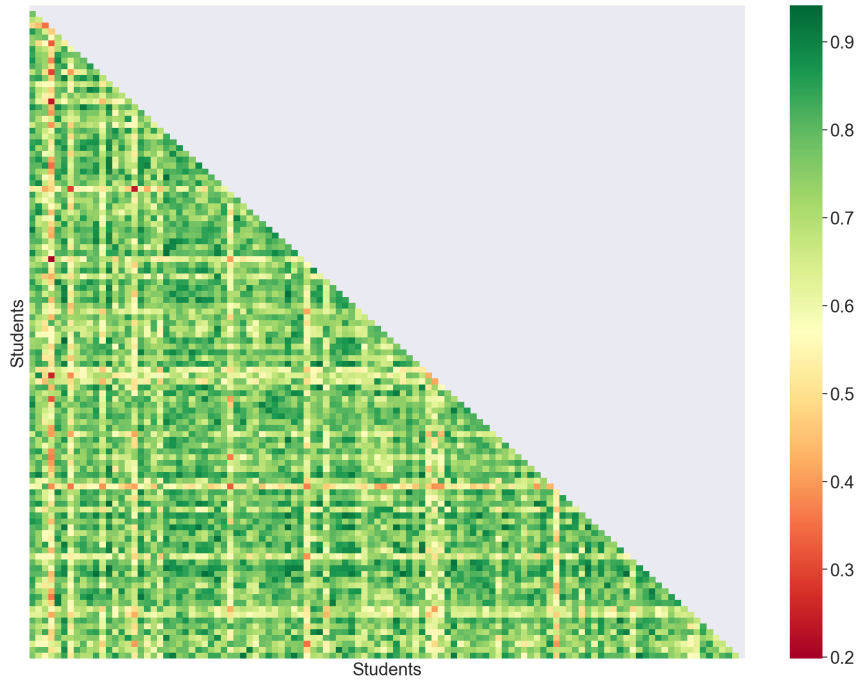


Figure 4.2: Uncleaned correlation matrix of students' transitions in Course#1-2018 dataset. The scale on the right side of the two figures indicates the range value of the correlation coefficients.

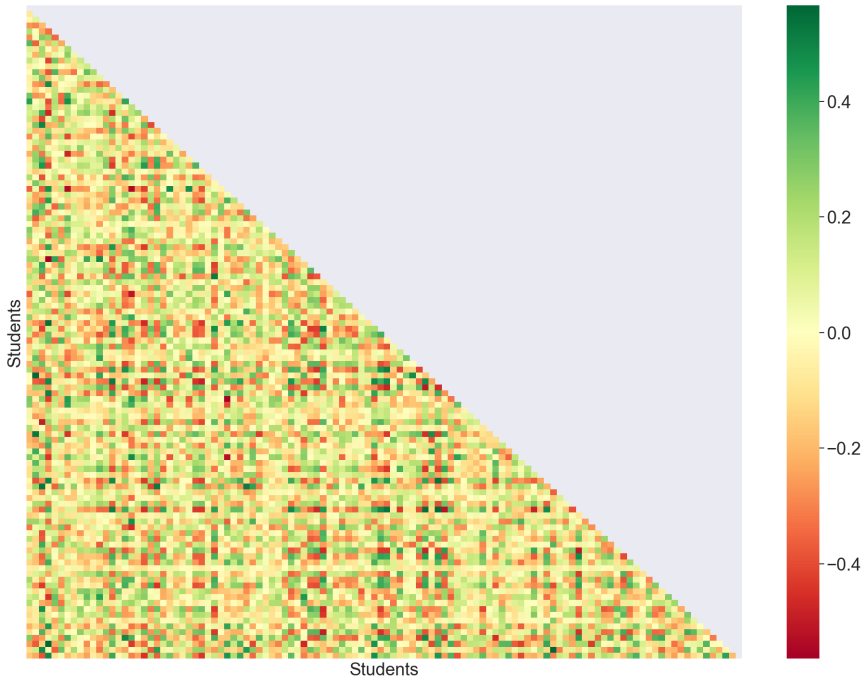


Figure 4.3: Cleaned correlation matrix of students' transitions in Course#1-2018 dataset. The scale on the right side of the two figures indicates the range value of the correlation coefficients.

In the following subsections, we discuss methods for cleaning the correlation matrix of a dataset as well as propose a method for cleaning the dataset based on Random Matrix Theory.

4.4.1 Cleaning the correlation matrix

Having reviewed a number of correlation cleaning methods (e.g *eigenvalue clipping* [98, 133] and *linear shrinkage* [80]), we adopt the *eigenvalue clipping* because it was found to be the best in terms of its ability of removing the noise while preserving the information part, i.e. as represented by the trace of the original correlation matrix. This approach simply utilises the results of the Marchenko-Pastur equation [33] instead of choosing a parameter during the cleaning process such as *linear shrinkage* and *Rotationally invariant, optimal shrinkage* [34]. The *eigenvalue clipping* provides robust out-of-sample performance [30] which has been widely adopted [160, 50].

Let $\lambda_1, ..., \lambda_N$ be the set of all eigenvalues of $\mathbf{C}$ and $\lambda_1 > ... > \lambda_N$, and i be the position of the eigenvalue such that $\lambda_i > \lambda_+$ and $\lambda_{i+1} \leqslant \lambda_+$.

Then we set

$$\lambda_j = 1/(N-i) \sum_{k=i+1}^{N} \lambda_k, \tag{4.10}$$

where $j = i+1, ..., N$. In other words, we keep all upper bound eigenvalues, i.e. those with information, and replace all lower bound eigenvalues, i.e. those within the bounds predicted by RMT, with their average value. Hence, we believe, this method can best preserve the trace of the original correlation matrix. The new set of eigenvalues can be used to construct a denoised eigenvalue and the spectrum associated with correlation matrix $\mathbf{C}_{denoised}$. [135].

The effect of the first eigenvalue and eigenvector can be removed from the denoised correlation matrix as follows [135], forming a cleaned correlation matrix as follows:

$$\mathbf{C}_{cleaned} = \mathbf{C}_{denoised} - W_1 V_1 W_1^T \tag{4.11}$$

where W_1 and V_1 are the first eigenvector and eigenvalue of $\mathbf{C}$, and ${W_1}^T$ is the transposed matrix of W_1. An example of the effect of the cleaned correlation matrix can be seen in Figure 4.2 and Figure 4.3. The two figures illustrate the correlation coefficients of the *transition-student data matrix* in the Course#1-2018 dataset, that is, each dot in the figures refers to the correlation of one student with another student. The scale on the right side of the two figures indicates the range value of the correlation coefficients. We also note that the diagonal of the matrix refer to the correlation of transition data of a student with her/himself (i.e. correlation value = 1).

It may be seen in Figure 4.2 that the majority of the dots are in different shades of green. This phenomenon may reflect a "trend effect", i.e. students' learning behaviours can be similar and highly positively correlated with other learners' behaviours in the same class. However, these issues may negatively influence the construction of prediction models. After cleaning the data, there are more neutral and dot shades of orange visible in Figure 4.3, indicating the negative correlation values. That is to say, the $\mathbf{C}_{cleaned}$ correlation matrix may show better differences in student learning behaviours, creating the higher chances to better cluster the students. Similar results are observed in all other datasets for Course#1 and Course#2. The use of the cleaned correlation matrix in Community Detection is discussed more in Section 4.5.

4.4.2 Cleaning the Dataset

While the cleaned correlation matrix is expected to be useful in Community Detection, the prediction of learning outcomes, however, requires a tabular dataset, as tabular data are necessary inputs for a large number of data mining methods, such as classification and clustering [147]. As a result, it would be necessary to clean the original data matrix instead of the correlation matrix to improve prediction models. In this section, we propose a method to clean the original data matrix based on Random Matrix Theory.

In terms of the eigenspectrum of the correlation matrix $\mathbf{C}$, let $\lambda_1, ..., \lambda_N$ be the set of all eigenvalues of $\mathbf{C}$ and $\lambda_1 \geq ... \geq \lambda_N$, and k be the position of the eigenvalue such that $\lambda_k > \lambda_+$ and $\lambda_{k+1} < \lambda_+$. In other word, $\lambda_1, ..., \lambda_k$ are outside the random matrix bands and $\lambda_{k+1}, ..., \lambda_N$ are inside. We note that λ_1 refers to the largest eigenvalues and the first principal component. The cleaned dataset $\widehat{\mathbf{G}}$ can be constructed as follows:

$$\widehat{\mathbf{G}} = \sum_{i=1}^{n} \mathbf{v}_i \mathbf{x}_i - \alpha \sum_{i=k}^{n} \mathbf{v}_i \mathbf{x}_i - \beta \mathbf{v}_1 \mathbf{x}_1 \tag{4.12}$$

where $\sum_{i=k}^{n} \mathbf{v}_i \mathbf{x}_i$ refers to the noisy part of the dataset based on RMT, and $\mathbf{v}_1 \mathbf{x}_1$ refers to the first principal component of data. The parameters $\alpha \in [0, 1]$ and $\beta \in [0, 1]$ control how much we want to remove the noise and trend parts from the original data, respectively. If α and β are equal to zero, Equation 4.12 is similar to Equation 4.9, and $\widehat{\mathbf{G}}$ reverts to $\mathbf{G}(n)$, i.e. the reconstruction of full original data from the principal component score dataset. If α and β have a unit value, we have a *fully cleaned dataset*. Otherwise, we have a *partly cleaned dataset* where both α and β are between 0 and 1. The cleaned dataset $\widehat{\mathbf{G}}$ can then be used as an input for machine learning predictive models. We expect that the performance of the predicting models using $\widehat{\mathbf{G}}$, either fully or partly cleaning will be improved in comparison with the use of the original dataset and simply PCA-based datasets.

4.5 Community Detection

This section briefly introduces the topic of Community Detection, followed by the details of the Community Detection methods used in this research.

4.5.1 Brief of Community Detection

Given a graph that contains a set of nodes and edges where each edge represents a connection between two nodes in the graph, the two following concepts can be defined:

- **Degree of a node:** the number of edges that connect the node to other nodes in the graph.

- **Community:** A set of nodes that are connected more densely to each other than to the rest of the graph [137]. Communities may or may not overlap with each other, depending on their definitions in specific applications.

- **Community structure:** The phenomenon of a graph whose nodes are organised into groups, called *communities* or *clusters* [137, 76]. A graph has a community structure if the nodes of the graph can be easily grouped into (potentially overlapping) sets of nodes such that each set of nodes is densely connected internally. In non-overlapping communities, the network divides naturally into groups of nodes with dense connections internally and sparser connections between groups. That said, the pairs of nodes are more likely to be connected if they are both members of the same community(ies), and less likely to be connected if they do not share communities.

- **Modularity:** A metric that is a commonly used function to measure the strength of the division of a graph into communities. The value range is normalised between -1 to 1. Modularity Q is defined as [119]:

$$Q = \sum_{c=1}^{n} \left[\frac{L_c}{m} - \left(\frac{k_c}{2m} \right)^2 \right] \tag{4.13}$$

  where:

  - the sum iterates over every community c in the graph;
  - L_c is the number of within-community edges for the community c, i.e., the edges that do not cross the border between communities;
  - m is the total number of edges;
  - k_c is the sum of degrees of the nodes in community c;

Graphs with high modularity have dense connections between the nodes within communities and sparse connections between nodes belonging to different communities.

- **Community Detection:** It refers to the procedure of detecting groups of interacting nodes in a graph based on its structural properties [137, 185, 94]. In other words, Community Detection can be considered as a clustering technique that can be applied to the graph to detect the communities with similar properties and behaviours so that they can be grouped. Many algorithms for Community Detection have been developed [99], which have been applied into the variety of disciplines such as biology and healthcare [188], Social Networks [20] and Economics [183]. With respect to the educational domain, the application of Community Detection has been limited and usually in the form of social network analysis. For example, a graph can be constructed based on the data about communications between students, e.g., asking questions and giving answers under each topic of the study [189], or discussions via learning forums [168] in online learning platforms, where students within the same community show a higher level of communication with each other than with students outside their community.

In the context of this research, we propose a novel approach based on the assumption that students with the same level of performance in study could show similarities in the learning behaviours. To verify whether students with similar behaviours, and vice versa, perform similarly on lab exams, we choose to adopt a graph-based approach. Generally, a graph is constructed based on the concept of a *distance matrix* discussed below. In the graph, each node represents a student and the edge weights between two nodes indicate the distance between learning behaviours of the two students. Then, Community Detection can be applied to the graph to detect the communities where students who have similar learning behaviours are grouped. The detected communities can be used for further analysis in terms of the relationship between their learning behaviours and outcomes.

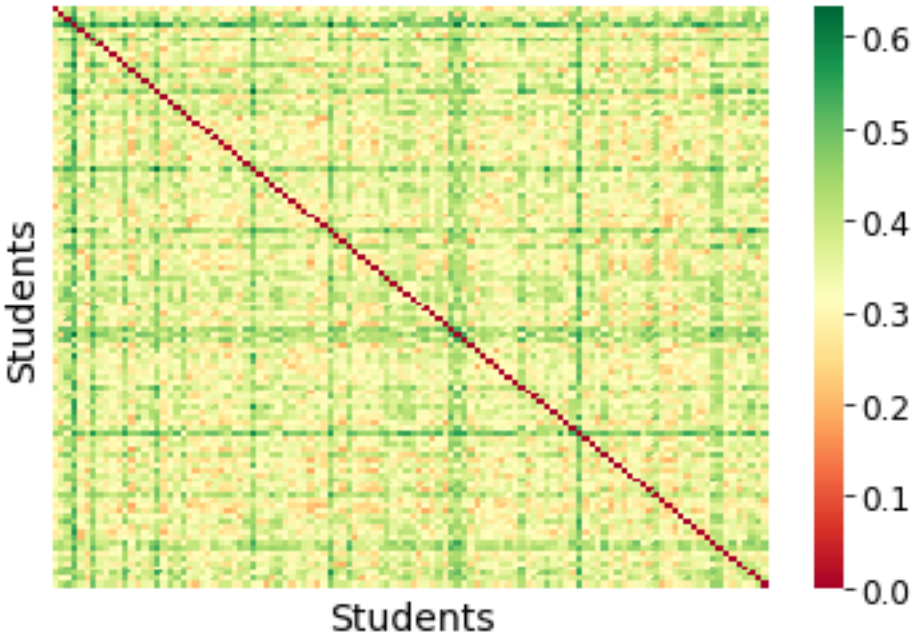


Figure 4.4: Learning Behavioural Distance matrix of students in Course#1 dataset.

4.5.2 Distance matrix of students learning behaviours

Although the correlation values appear to be useful in reflecting the similarity and difference in students' learning behaviours as mentioned in Section 4.3, they are not appropriate metrics as they do not satisfy the non-negativity and triangle inequality conditions [135]. For example, the difference between the correlations (0.8, 1.0) is the same as (0.1, 0.3), but the former tuple illustrates a higher difference regarding codependence. Fortunately, it is possible to translate the correlation matrix into a distance matrix $\mathbf{D}$ as follows [135]:

$$\mathbf{D}_{ij} = \sqrt{0.5 * (1 - \mathbf{C}_{ij})} \tag{4.14}$$

with $D_{ij} \in [0, 1]$ where D_{ij} is a distance value of learning behaviours between the two students i and j. The value closer to unity refers to two students interacting completely differently with the course material items, while the value closer to zero indicates that two students behave similarly. Figure 4.4 indicates an example of the learning behavioural distance matrix of students in Course#1-2018 dataset.

The diagonal comprises of zero values, illustrating the behavioural distance between each student and him/herself. The other distance values range from zero to more than 0.6. The distance matrix can be used to construct a community graph which is discussed in the next section.

4.5.3 Constructing a graph from the distance matrix

A graph can be constructed directly using the distance matrix values D_{ij} as edge weights. Unfortunately, such a network is hardly a readable weighted complete graph as each node (student) has a connection to all other nodes in the graph. Additionally, we note that the time complexity of community detection algorithms is proportional to the number of edges and nodes in the graph [192]. For example, the time complexity of the Girvan-Newman algorithm [76], which is one of the most commonly used Community Detection techniques (see more details of Girvan-Newman algorithm in Section 4.5.4), is $O(m^2n)$ where m is the number of edges and n is the number of nodes (or students). For such a fully connected graph, the number of edges is $m = n(n-1)/2$, which, in the worst case, may lead to the time complexity of the algorithm of $O(n^5)$. To overcome this issue, one possible solution is to reduce the number of edges in such a fully-connected graph. It is important to minimise the number of edges in the constructed graph while preserving the purpose of grouping students having similar behaviours.

In the context of our research, all the values of the distance matrix of each dataset are different to each other. In other words, all edge weights of the fully connected graph constructed from the corresponding distance matrix are unique. Taking advantage of this characteristic, we adopt the notion of Minimum Spanning Tree (MST) [179], i.e. an MST is constructed for each graph and it connects all students in a course without having any loops. With the distance matrix $\mathbf{D}$ as the adjacency matrix of a graph, an associated MST is constructed such that the sum of all edges in the graph is minimal for all possible spanning trees. For example, an MST can be constructed using the Kruskal's algorithm [97] as follows:

- Step 1. Sort all edges in the graph in ascending order of their edge weights.
- Step 2. Select the smallest edge from the unselected edges
- Step 3. Check if the new selected edge forms a cycle or loop in a spanning tree. If the edge does not form a loop/cycle, then include that edge in the MST. Otherwise, discard the edge.
- Step 4. Repeat Steps 2 and 3 until all nodes are connected.

We note that if all edge weights of a graph are unique, then the graph has only one corresponding MST. Hence, in our case, each course dataset can be used to produce a single associated MST. It can be seen that the MST of a set of n students is a graph with $n-1$ edges, reducing the time complexity of the Girvan-Newman algorithm to $O((n-1)^2 n) = O(n^3)$, see [155] for more detail. Furthermore, in a MST constructed from learning behavioural datasets, each student can be connected to one or more other students who have the most similar learning behaviours with that student, which is based on the premise that the distance matrix measures the similarities in learning behaviours between students. In this way, the clustering purpose is preserved.

4.5.4 Community detection on MST graph

Based on the MST constructed from the distance matrix, it is possible to advance to the next step by clustering communities whose elements have similar characteristics, i.e. students having shorter distances of learning behaviours are clustered into a community. This process can be supported by several community detection methods [27, 76]. In this research, we utilise the popular detection algorithm from Girvan-Newman [76]. We also compare the results produced by Girvan-Newman with those of the Louvain algorithm [27] to ensure the stability of the community analysis. Both algorithms are explained below.

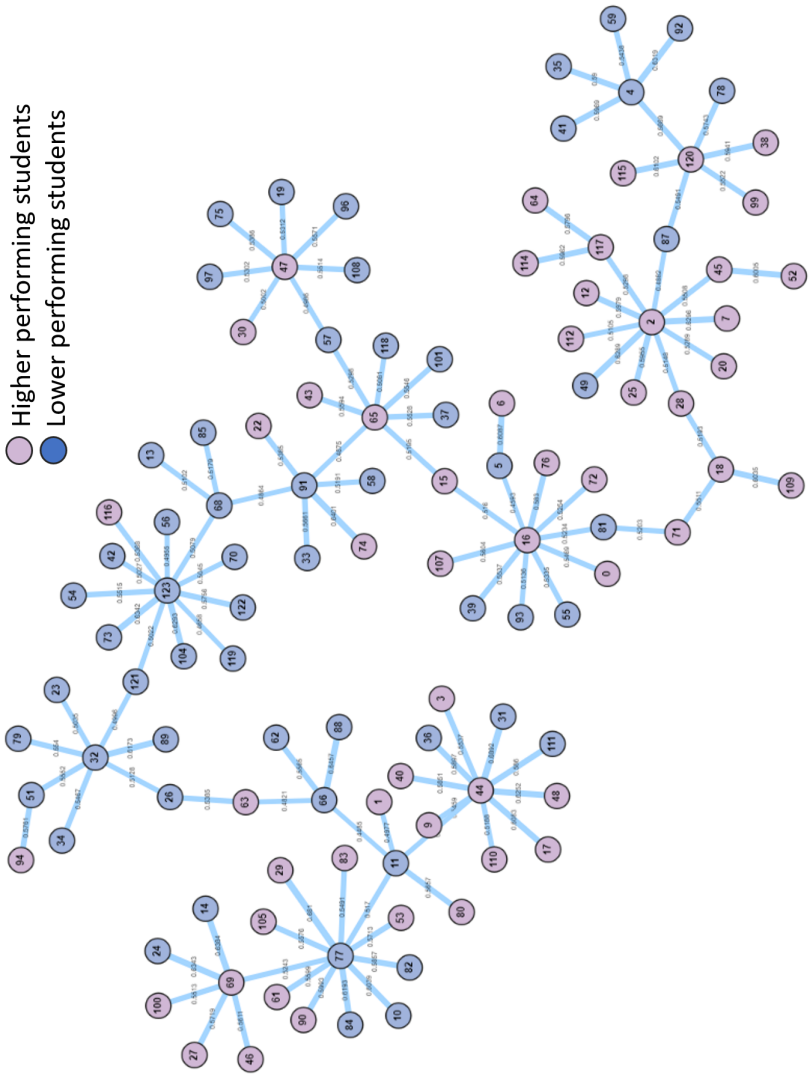


Figure 4.5: An example of an MST constructed from a distance matrix of the Course#1-2018. Purple nodes refer to the higher performing students while Blue nodes refer to the lower performing students. Each edge corresponds to the distance of learning behaviours between two students, as per 4.14

Background of the Girvan-Newman algorithm

In this thesis, we utilise Girvan-Newman algorithm, which has been applied in various domains such as biology [188], finance and cryptocurrencies [43], as a demonstration of using community detection in analysing students' learning behaviours.

The main idea of the Girvan-Newman algorithm is to find the edges in a network that occur most frequently between other pairs of nodes by finding edge betweenness. *Edge betweenness* is the number of the shortest paths between pairs of nodes that pass through the edge of the original network [76]. It is expected that the edges joining communities have a high edge betweenness. The detected community structure of the network will be much more dense when the highest betweenness edges are eliminated. Therefore, the nodes in the detected community are more separate from the other nodes in the network. In other words, the algorithm aims to divide the whole network into smaller communities or groups by progressively removing the edges with the highest *edge betweenness* until no edges remain. The Girvan-Newman algorithm can be divided into four steps as follows [76]:

- Step 1. Compute the edge betweenness centrality for every edge in the graph.
- Step 2. Eliminate the edge with the highest betweenness centrality found.
- Step 3. Compute the betweenness centrality for every remaining edge in the graph.
- Step 4. Repeat Steps 2 and 3 until there are no more edges left to be eliminated.

It should be noted that we take into account the weight of edges when calculating the edge betweenness. The nodes, (i.e., students) in a smaller community are more connected to each other than the students outside the community. Figure 4.5 illustrates an example of the MST constructed from the data for Course#1-2018. The detected groups can be used for further investigation regarding their performance in lab exams. In particular, we can use statistical tests to verify if the lab exam grades are significantly different between the communities. As it is not guaranteed that the

data of students in each community will be distributed normally, a nonparametric test is preferred in this case, i.e. Mann-Whitney U Test [117] has been used. It is also possible to verify if the two communities interacted differently with each course material item in the learning system. Further research is discussed in Section 5.3.

Background of the Louvain algorithm

Louvain algorithm was proposed for the extraction of the community structure from large networks in [27] and can be seen as one of the most commonly used Community Detection techniques [170].

The algorithm is developed based on the concept of modularity optimisation and follows a heuristic approach. In particular, the algorithm is implemented as follows:

Step 1. Each node is assigned to a different community. As a result, the number of communities is equal to the number of nodes.

Step 2. For each node i, the node is assigned to the community of its neighbour j. The gain in modularity ΔQ is calculated using the following formula:

$$\Delta Q = \left[\frac{\sum_{in}+k_{i,in}}{2m} - \left(\frac{\sum_{tot}+k_i}{2m}\right)^2\right] - \left[\frac{\sum_{in}}{2m} - \left(\frac{\sum_{tot}}{2m}\right)^2 - \left(\frac{k_i}{2m}\right)^2\right] \tag{4.15}$$

where:

- $\sum_{in}$ is the sum of the weights of the edges within the community C;
- $\sum_{tot}$ is the sum of the weights of the incoming edges of the nodes in the community C;
- k_i is the sum of the weights of the incoming edges of node i;
- $k_{i,in}$ is the sum of the weights of the outgoing edges of node i that point to the community C;
- m is the sum of the weights of all edges in the network;

If the gain of modularity ΔQ is positive, node i is moved to the community of its neighbour j. Otherwise node i will stay in its own community. The first step stops when there is no further gain of modularity ΔQ by moving any nodes in the network.

Step 3. New nodes, called *super-nodes*, will be formed from identified communities, i.e., all nodes from a community are merged into a *super-node*. The edge weights between the new *super-nodes* are calculated by summing the edge weights between the nodes of the two identified communities. In addition, each super-node can have a self-loop with a weight equivalent to the sum of the edges' weights between the nodes of this community.

Step 4. Repeat Steps 2 and 3 until no further gains in modularity ΔQ are detected.

Although we mainly focus on the Girvan-Newman algorithm, in the scope of this work, the Louvain algorithm [27] is also used as a benchmark to verify if the two algorithms produce significantly different results. Particularly, we utilise the v-measure score [153], a widely-used clustering metric to measure the agreement of two independent community detection strategies produced by the two algorithms (i.e., Girvan Newman and Louvain) for each dataset. The value of the v-measure score falls between 0 and 1. The higher the value is, the higher level of agreement between the results produced by the two community detection algorithms is. The calculation of v-measure score is also supported by the *sklearn* library [128].

Furthermore, we also investigate whether our cleaning method can support the Louvain method to generate better communities with lower *mixed community rates* for the cleaned data compared to the original data. The concept of *mixed community rates* is introduced below.

Selecting the number of detected communities

We observe that the Girvan-Newman algorithm can be seen as a hierarchical method, i.e., it constructs a dendrogram that shows a hierarchical clustering structure. The number of detected communities can, therefore, range from unity to the number

of nodes in the graph where each community contains only one node. When using Girvan-Newman, it is necessary to determine the criteria to decide the cut-off level in the dendrogram to create the resulting communities.

In this research, we define the concept of a *mixed community rate* as follows: Let $C = (c_1, c_2, ..., c_n)$ be a community structure, $c_i = (h_i, l_i, n_i)$ be a detected community where h_i is the number of higher performing students, l_i be the number of lower performing students in the community c_i. The label n_i of the community c_i is identified as Equation 4.16 below:

$$n_i = \begin{cases} \text{higher-performing}, & \text{if } h_i/(h_i + l_i) \geq k \\ \text{lower-performing}, & \text{if } l_i/(h_i + l_i) \geq k \\ \text{mixed}, & \text{otherwise} \end{cases} \tag{4.16}$$

The parameter k can be configured depending on the purpose of the analysis. In the ideal case of $k = 1$, a community will only be labelled as higher or lower performing if it contains only higher or lower performing students. However, we expect the similarity in learning behaviours between students in practice, and it could be difficult to detect such a homogeneous community. Instead, we set $k = 0.7$, i.e., a community is labelled "higher-performing" if there are greater than or equal to 70% of higher performing students in the community and similarly for "lower-performing" communities. Otherwise, if the number of higher performing students is in between 30% and 70%, the communities are labelled as "mixed". The *mixed community rate* of a community structure can be computed as follows.

$$\text{mixed community rate} = \frac{\text{No. of mixed communities}}{\text{total no. of communities}} \tag{4.17}$$

Higher/lower performing communities may include key features about student success, while mixed communities may contain less information. As a result, we expect a good community structure to contain fewer *mixed communities*. Based on the

mixed community rate indicator, it is possible to investigate each possible community in the resulting dendrogram from the Girvan-Newman algorithm and identify the number of detected communities by considering their *mixed community rates*. We have also made a comparison between the original dataset and the cleaned dataset in terms of the community structures detected from them. If cleaned datasets can be used to produce a community structure with lower *mixed community rates*, the cleaning method can highlight its effectiveness in community analysis.

4.6 Machine Learning Prediction techniques

One of the objectives in this research is to investigate the predictability of learning behaviour data of students for their learning outcomes. A number of classification techniques have been applied to build predictive models that can detect whether a student is likely to belong to the higher or lower performing cohort. This section gives a brief introduction about the machine learning classification tools used in this research, along with the metrics to evaluate the quality of the trained models.

4.6.1 Classification algorithms and tools

In terms of prediction algorithms, Support Vector Machine (SVM) has been reported to be the most effective general technique for the data captured from MOOCs in many contexts [52, 158]. In addition to SVM, for reference and comparison purposes, we also choose four additional classification techniques including XGBoost [45], Logistic Regression [32], Gradient Boosting [67] and K Nearest Neighbours [55] due to their widely applications in Learning Analytics domain [115].

A summary of classification techniques and tools which are used in this thesis is given below:

- Support Vector Machine (SVM) aims to form a hyperplane with the maximum margin from both classes of data [51]. In this study, the hyperplane is expected

to separate the 'higher performing' and 'lower performing' student cohorts based on their frequency of interactions with course learning items.

- Boosting [67] is a supervised machine learning algorithm used for classification and regression problems. It is an ensemble technique with the main objective to minimise the loss function by adding weak learners using a gradient descent optimisation algorithm. In other words, Gradient Boosting uses multiple weak learners to produce a strong model for classification.

- XGBoost [45] stands for "Extreme Gradient Boosting", and can be seen as an optimisation of other Gradient Boosting techniques. XGBoost was to focus on the improvement of speed and performance of the training process of the models. In this research, we use the *XGBoost* library in [44].

- Logistic Regression [32] can be seen as a commonly-used and reliable prediction method in educational scenarios [63, 31]. It estimates the probability of a categorical variable from a number of features [118].

- K Nearest Neighbours (KNN) [55] is a predicting method which aims to build a non-parametric classifier. KNN algorithm uses data with several classes to predict the category of the new sample point.

These methods can be used by educators to predict student outcomes, i.e. classify whether a student is likely to fall into the higher or lower performing class. This could help to identify "at-risk" students during the study period. In terms of development tools, we use *sklearn* libraries [128] and Python as the main programming language.

4.6.2 Evaluation of predicting models

In this research, we aim to build binary classification models where there are only two possible output classes, i.e., a higher and a lower performing student. In order to evaluate the predictive power, as well as, to compare the performance between

		Actual value	
		Positive	**Negative**
Predicted value	**Positive**	TP True Positive	FP False Positive
	Negative	FN False Negative	TN True Negative

Figure 4.6: Confusion Matrix

the models, a number of evaluation metrics have been used. In particular, we utilise *Accuracy*, *F1-Score* and *ROC-AUC* [87].

When a binary classifier is used to predict the class of each data point in a test set, the result falls into one of the following four categories:

- True Positive (TP) is the result where the classifier correctly predicts the positive class;
- True Negative (TN) is the result where the classifier correctly predicts the negative class;
- False Positive (FP) is the result where the classifier incorrectly predicts a sample as positive, but it actually belongs to the negative class;
- False Negative (FN) is the result where the classifier incorrectly predicts a sample as negative, but it actually belongs to the positive class;

The Confusion Matrix can be considered as a performance measurement for the classification in machine learning models where the output can be two or more classes. It forms a table with the combinations of actual and predicted values, with the values of TP, FP, FN and TN delivered by a classifier. Figure 4.6 illustrates

a confusion matrix for binary classification. This is important to calculate the evaluation metrics below:

- *Accuracy* [87] aims to calculate the proportion of cases that are predicted correctly by the classifier. Accuracy metric is calculated as follows:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.18}$$

  A limitation of the *Accuracy* metric is that if the training set is unbalanced, the model is likely to return the dominant class and *Accuracy* may, therefore, give a misleading sense of the quality of the model.

- *Precision* [87] aims to calculate the proportion of the correctly predicted cases that are actually positive, which can be calculated as follows:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{4.19}$$

- *Recall* [87] aims to calculate the proportion of the actual positive cases that the model can predict correctly, which can be calculated as follows:

$$\text{Precision} = \frac{TP}{TP + FN} \tag{4.20}$$

- *F1-score* [87] can be seen as the combination between Precision and Recall metrics:

$$\text{F1-score} = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{4.21}$$

- *ROC-AUC*: The ROC-AUC curve is a performance measurement for classification. It explains how much the model can distinguish between classes. Details of ROC-AUC metric can be found in [88]. In our research, the higher the value of ROC-AUC is, the better the model can predict an actual higher performing student as higher performing and an actual lower performing student as lower performing.

The value range of the metrics *Accuracy*, *F1-score*, and *ROC-AUC* is between 0 and 1. The higher value is, the better the model can predict the data.

4.7 Conclusion

In this chapter, we have introduced the methods and techniques that have been used in this thesis. The experiments have been carried out using the learning event logs collected from our bespoke learning system for students in the two programming courses (namely Course#1 and Course#2) over the three academic years, i.e., 2018, 2019 (pre-COVID19) and 2020 (during COVID19), forming six datasets for each course in each year. From the log data, the features for students' learning behaviours are extracted, generating behavioural datasets. Students are also classified into two groups, that is, higher-performing and lower-performing cohorts based on grades on the final lab exam of each course.

We have proposed a novel approach to clean the effect of noise and trend in the extracted behavioural datasets, utilising the characteristics of Random Matrix Theory. Then Community Detection techniques (i.e., Girvan Newman and Louvain algorithms) have been implemented on the extracted behavioural datasets to identify the communities of students with similar behaviours and to verify if their learning outcomes are the same. The ability to use learning behavioural data to predict students' learning outcomes has also been tested with commonly-used machine learning classification methods.

Experimental results of the applications of these methods on the datasets are reported and discussed in the following Chapters.

Chapter 5

Experimental Results

5.1 Introduction

The solution for *Sub RQ1* has been proposed in Chapter 4 - Section 4.2 regarding the storage and extraction of students' behavioural data as well as Section 4.2 and 4.3 with respect to the processing of the data, i.e. cleaning the effect of noise and trend. Accordingly, this chapter shows the experimental results of the analysis of the collected learning behavioural log data using our research approaches. The results are expected to address the research questions outlined Chapter 1. Section 5.2 and 5.3 are expected to provide answers for *Sub RQ2* and *Sub RQ3* in terms of the relationship between the groups of students who have similarities in their learning behaviours and academic performances. Section 5.4 indicates experimental results on the predictability of the models constructed from the behavioural datasets, addressing *Sub RQ4*. In each section, we describe which datasets are going to be used and which methods are applied in the analysis, followed by the analysis results. Evaluation of the research methods is also discussed in the sections below.

Table 5.1: Detail of the datasets for Principal Component Analysis (PCA)

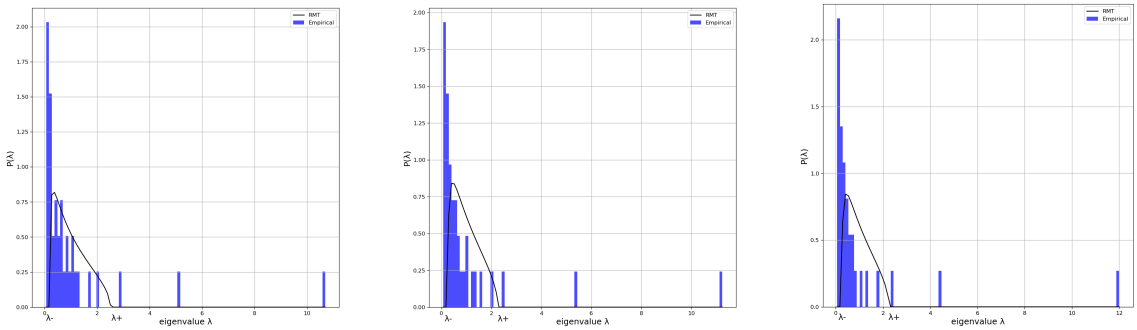
Dataset	Number of students	No. of learning material items	Number of Higher performing students	Number of Lower performing students
Course#1-2018	112	37	54	58
Course#1-2019	151	37	87	64
Course#1-2020	128	31	69	59
Course#2-2018	62	28	44	18
Course#2-2019	48	29	33	15
Course#2-2020	65	34	52	13

5.2 PCA for Students' Learning Behaviours using single event frequency features

5.2.1 Data and method

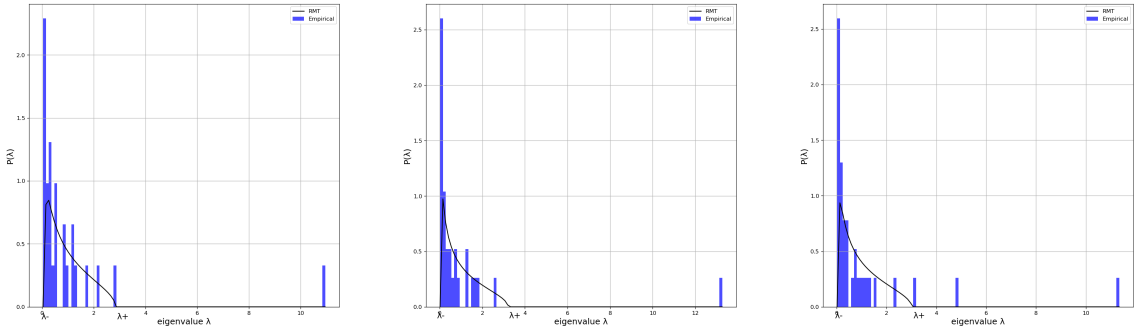
In this section, we extract the *single event frequency features* from the event logs of Course#1 and Course#2 over the three academic years (ie. 2018, 2019 and 2020). This forms six datasets for the Principal Component Analysis, namely Course#1-2018, Course#1-2019, Course#1-2020 and Course#2-2018, Course#2-2019 and Course#2-2020. In each dataset, students are classified into two cohorts based on their exam results, i.e., "Higher-performing" and "Lower-performing". A summary of the extracted datasets can be seen in Table 5.1.

For each dataset, we calculate its correlation matrix, eigenvalues and the corresponding eigenvectors, as well as estimating the principal components. Briefly, from a m x n dataset where m is the number of rows and n is the number of columns, n eigenvalues can be computed, corresponding to n principal components. Each eigenvalue/principal component is pair associated with an eigenvector of size n, i.e., a vector that contains n elements. From this, the data values, eigenvalues and eigenvectors can be used to calculate the scores of the principal components. Details of the PCA method are described in Section 4.3. The summary of results is discussed in Section 5.2.2 below.



(a) Course#1-2018 (b) Course#1-2019 (c) Course#1-2020

Figure 5.1: Distribution of empirical eigenvalues and theoretical eigenvalues predicted by RMT for the datasets of Course#1

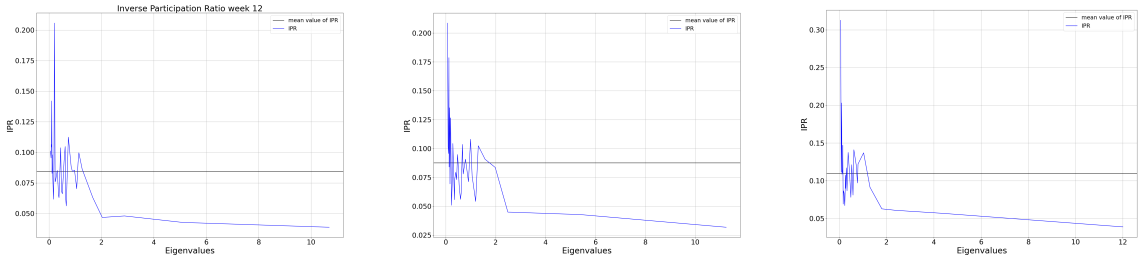


(a) Course#2-2018 (b) Course#2-2019 (c) Course#2-2020

Figure 5.2: Distribution of empirical eigenvalues and theoretical eigenvalues predicted by RMT for the datasets of Course#2

5.2.2 Selecting the key information part from the dataset

Figures 5.1 and 5.2 illustrate the probability distribution of empirical eigenvalues based on the six extracted datasets mentioned in Subsection 5.2.1. The figures also compare the empirical eigenvalues with the theoretical eigenvalues predicted by RMT for the random matrix that has the same Q-factor ratio, i.e., the Q-factor ratio between the number of rows and columns, with each dataset (see Sections 2.3.3 and 4.3 for more detail about Random Matrix Theory). In general, for all datasets, the majority of empirical eigenvalues fall within the range between the largest (λ_+) and smallest eigenvalues (λ_-), which are distributed within the black curve in all figures. For example, in Course#1-2018, 34/37 (91.9%) empirical eigenvalues are within the range [$\lambda_- = 0.18$, $\lambda_+ = 2.47$], as represented in Sub-figure 5.1a. Similar



(a) Course#1-2018 (b) Course#1-2019 (c) Course#1-2020

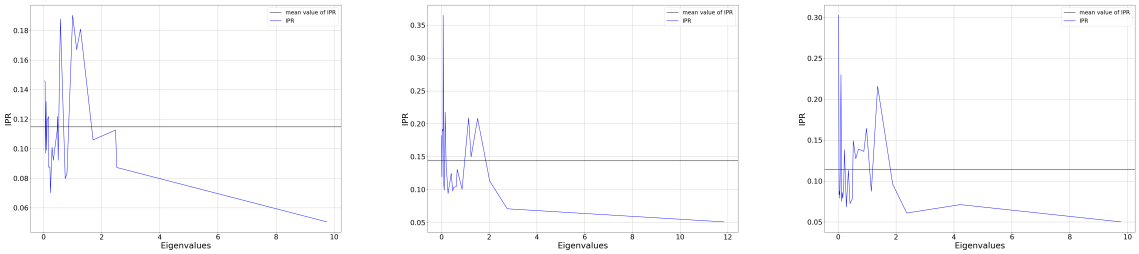
Figure 5.3: Inverse Participation Ratio of the empirical eigenvalues for the datasets of Course#1

results can be found in the remaining subfigures in Figure 5.1 and 5.2c. This is in agreement with previous studies which have found that a large proportion of empirical eigenvalues were predicted by RMT [56].

These observations indicate that there is a measure of randomness in the Eigen spectrum of each dataset. That is to say, the majority of eigenvalues would appear to merely follow a random pattern. The remaining eigenvalues, which are higher than the upper limit λ_+, are outside the noise area and have the potential to contain key information about the dataset, i.e., information on the learning behaviours of the students in the corresponding Course. As a result, these eigenvalues, which are associated with principal components, can be selected and used for further analysis as they are expected to reveal insights about the dataset. In particular, based on Figure 5.1, the first three principal components in each Course#1 dataset can be selected. On the other hand, in Course#2, only the first principal component can be selected for both Course#2-2018 and Course#2-2019 datasets, while Course#2-2020 yields three principal components (Figure 5.2c). We refer to all selected principal components as *significant principal components*.

In addition, we can measure the contribution of each data column to the principal components by using the Inverse Participation Ratio, which is described in Section 4.3. Figures 5.3 and 5.4 show the IPR values for the eigenvalues of the correlation matrix of each dataset. In each sub-figure, the x-axis refers to the empirical eigenvalues for the corresponding dataset, and the y-axis refers to the IPR values.

Based on the IPR values, it is possible to calculate the value 1/IPR for each



(a) Course#2-2018 (b) Course#2-2019 (c) Course#2-2020

Figure 5.4: Inverse Participation Ratio of the empirical eigenvalues for the datasets of Course#2

eigenvalue. The higher the value of 1/IPR is, the greater the number of eigenvector elements contribute to explaining the variance of the associating principal component. Hence, both Figures 5.3 and 5.4 indicate a pattern where the first few principal components, i.e., the PCs with eigenvalues are greater than λ_+, in each dataset are explained by the majority of the data columns. For example, in Sub-Figure 5.3a, the number of eigenvector elements that significantly contribute to the 1st, 2nd and 3rd components are 30, 23 and 23 out of 37, respectively. We note that each eigenvector element refers to a course learning item (e.g., Labsheet_1, Lecture_2, Practice_3 etc.), indicating how much the learning item contributes to the Principal Component. With the high number of learning items contributing to the principal components, this finding implies that the students appear to access and use most of the course material items delivered during the courses.

5.2.3 Student's Learning Behaviour before the COVID19 pandemic

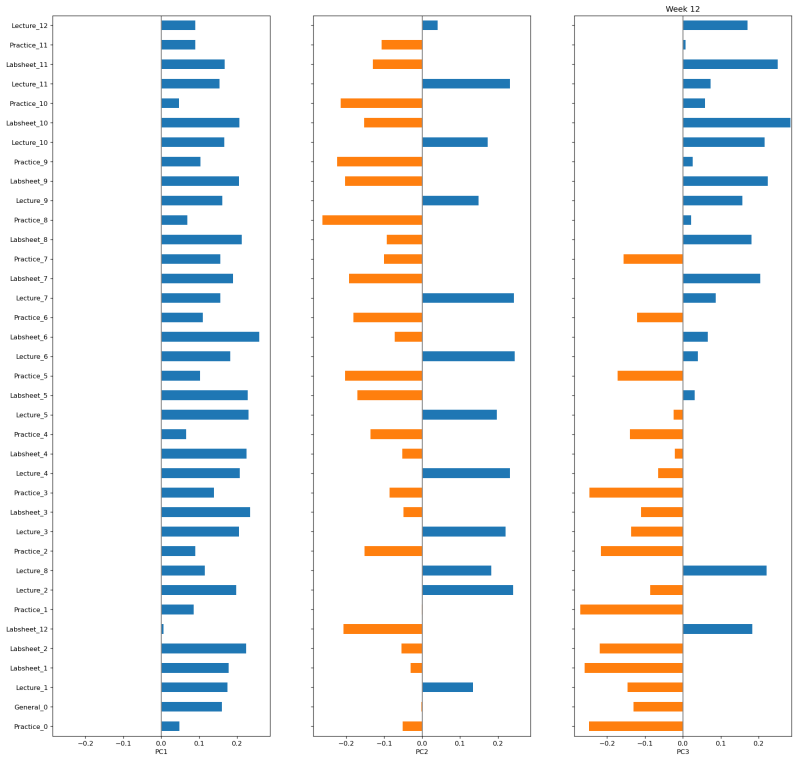
Firstly, we focus on the loadings of the selected principal components, (i.e., eigenvector elements of the dominant eigenvalues), of the datasets for Course#1 that were delivered before the COVID19 pandemic (2018 and 2019). The result is represented in Figure 5.5.

With respect to the first principal component (PC1), all component loading values are positive in both datasets (Figure 5.5a-PC1 and 5.5b-PC2). Such a positive

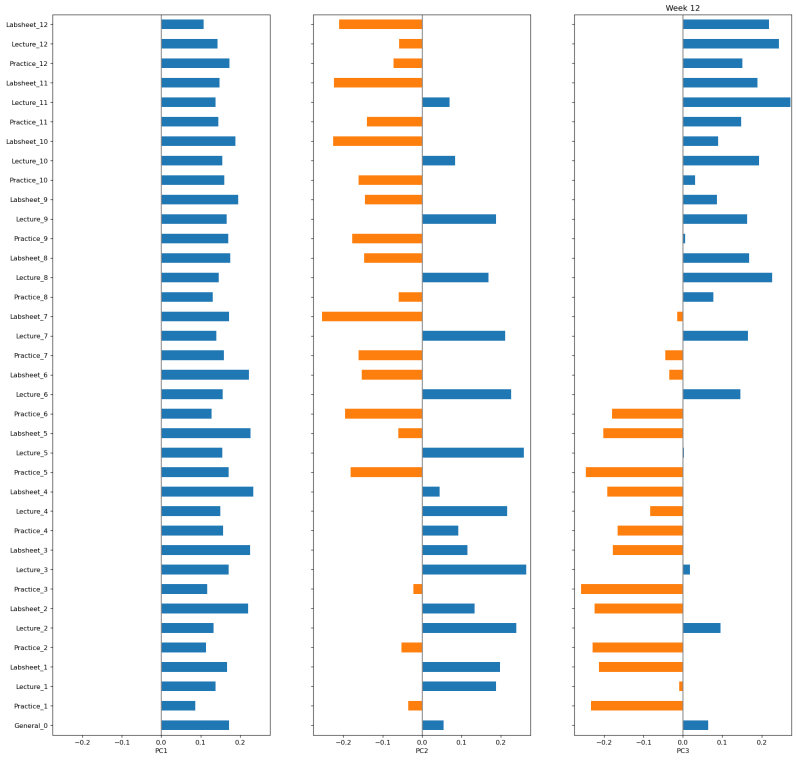
value of a component indicates that the component is positively correlated with the corresponding principal component scores. It is possible to say that students generally used almost all learning material items provided throughout the study. Furthermore, when we take into account the scores of PC1, there is no statistical difference between the PC1 scores for the higher and lower-performing cohorts (*t-test p-value* = 0.3 for Course#1-2018 and 0.46 for Course#2-2019, which are both more than 0.05). The statistical testing illustrates the similarity in the learning behaviours of students in the class. This indicates that almost all students have similar behaviours with the course material items during most of the semester. They participated in learning activities and followed the instructions and requirements given by the lecturers. Hence, we would argue that the first principal component in both datasets for Course#1-2018 and 2019 may indicate the *trend* effect on the students' learning behaviours in both courses. We, therefore, label PC1 as the "trend dimension". This finding reflects the fact that students were participating in a structured module, i.e., they mostly followed a designated timetable and a similar learning pathway in the classes.

On the other hand, the loadings of PC2 in both years of Course#1 indicate the difference between the students' learning behaviours in different types of learning material items. Particularly, from PC2 in both years of Course#1, the loadings for Lecture items show positive values while the sign of the values for practice-related items (i.e., Practice and Labsheet) are negative. In other words, there is a contrast between learning item types, i.e., lecturing and practice-related learning items in terms of students' interactions. This observation indicates that different students might interact differently with the learning item types. Students who focus more on solving programming exercises seem to have fewer activities on reading lecture notes and vice-versa. For this reason, we label PC2 as the "learning material item dimension".

In addition, PC3 in both years of Course#1 shows evidence of the difference in the students' interactions with learning material items at different times during the semester. Particularly, the sign of the values for learning material items that were



(a) Course#1-2018



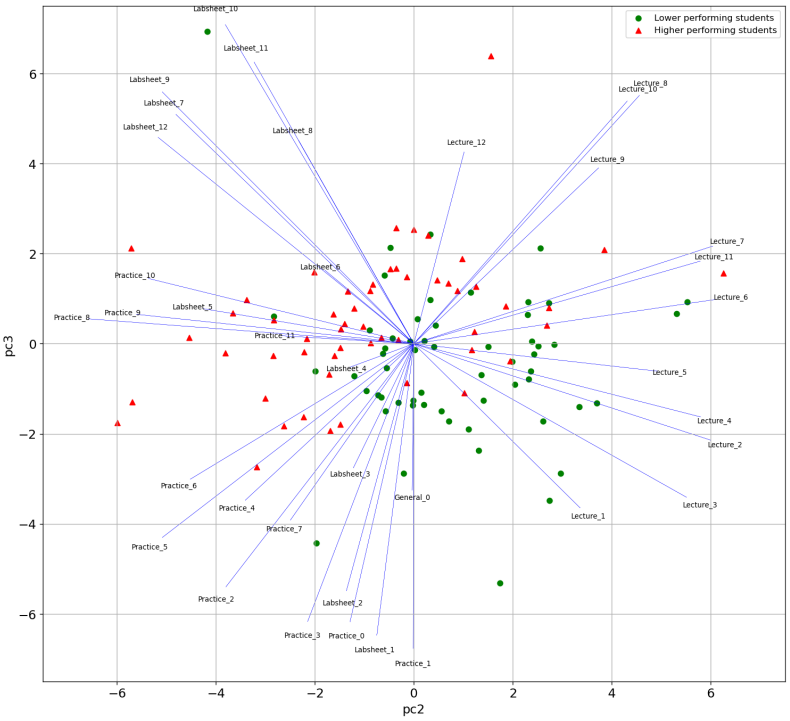
(b) Course#1-2019

Figure 5.5: The loadings (eigenvector components) of the PC1, PC2 and PC3 extracted from the Course#1 datasets for the year of 2018 and 2019 (before COVID19). The blues refer to positive values and the oranges refer to negative values

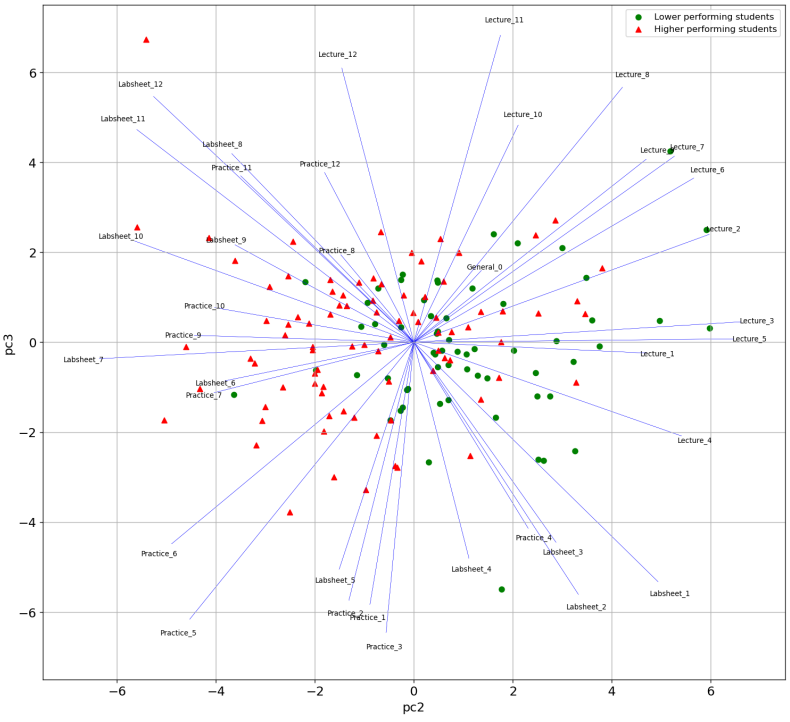
delivered at the beginning of the semester tend to be different from that for the items delivered at the end of the semester. This result may refer to the fact that students could be more or less focused on using learning material items in the first few weeks in comparison with the rest of the semester. Due to the difference, we label the PC3 as "time dimension".

In contrast to PC1 which merely shows the "trend effect" or the similarity in students' learning behaviours, PC2 and PC3 also appear to illustrate the dissimilarity between the behaviours of the two cohorts (higher and lower performing) of the students. In order to detect the dissimilarity, we analyse the combination between principal components' loadings and scores by creating biplots, a visualisation technique displaying information on both samples and variables in the data matrix [78]. The biplots represent the second and third principal components (PC2 and PC3) loadings and scores of both years (2018 and 2019) of Course#1 and can be seen in Figures 5.6. Each green or red dot refers to the PC scores of a student in the dataset. The green dots represent lower performing students while the red dots represent higher performing students. Additionally, the blue lines demonstrate the loadings of the PC2 and PC3, where each loading line implies how much the corresponding learning item contributes to the PC2 and PC3 scores. If a dot is located on the same side with a loading line along a principal component in the graphs, (i.e., the sign of the score and loading values are the same), it implies that there is a positive correlation between the corresponding student's data and feature (learning material item), and vice versa if a dot is located on an opposite to a loading line.

Both subfigures in Figures 5.6 show a common pattern in both PC2 and PC3. Regarding PC2, it can be seen that the data for most students in the lower performing cohort (green dots) are located on the same side as the loadings of the Lecture items, which indicates a positive correlation. Meanwhile, the majority of students in the higher performing cohort (red dots) show a positive correlation with practice-related items. That is to say, in both years of Course#1, higher performing students show a correspondingly high interest in doing programming exercises, which is expected. Higher performing students are able to adopt the new concepts



(a) Course#1-2018



(b) Course#1-2019

Figure 5.6: The biplot of the PC2 and PC3 extracted from the Course#1 datasets for the year of 2018 and 2019 (before COVID19).

in the lecture notes faster which allows them to focus on practice rather than to constantly revise lecture notes. Good students could also try to solve a problem in various ways, which increases their interactions with practice-related items. In contrast, lower performing students were likely to stick to reading lecturing documents. One reason for this could be that it was more difficult for these students to understand the theory delivered in the lecture, which led to the students go back to revise the lecture notes. This can be demotivating for students when they are trying to solve given programming exercises and labsheets.

Regarding Course#2 datasets, only the first principal component is significant and selected in both years (Course#2-2018 and 2019). All loadings of these components have positive values, which is similar to the result for Course#1. Therefore, it is possible to say that the "trend effect" is also represented in Course#2. However, there is not enough evidence to detect any difference in students' learning behaviours due to the insignificance of the other principal components (e.g. PC2 and PC3). As a result, we conclude that the students show similar study patterns when interacting with learning material items during the semester in Course#2 before the COVID19 pandemic. The dataset could be influenced by the fact that Course#2-2018 and 2019 were given to business informatics students who may have less motivation for programming. The content and exams of Course#2 were also less challenging comparing to Course#1 which is designed for Software Engineering students with a higher level of difficulty expectation. Therefore, it would be easier for Course#2 students to achieve desirable grades regardless of the level of their programming skills and learning, leading to no difference in the learning behaviours of the student cohorts.

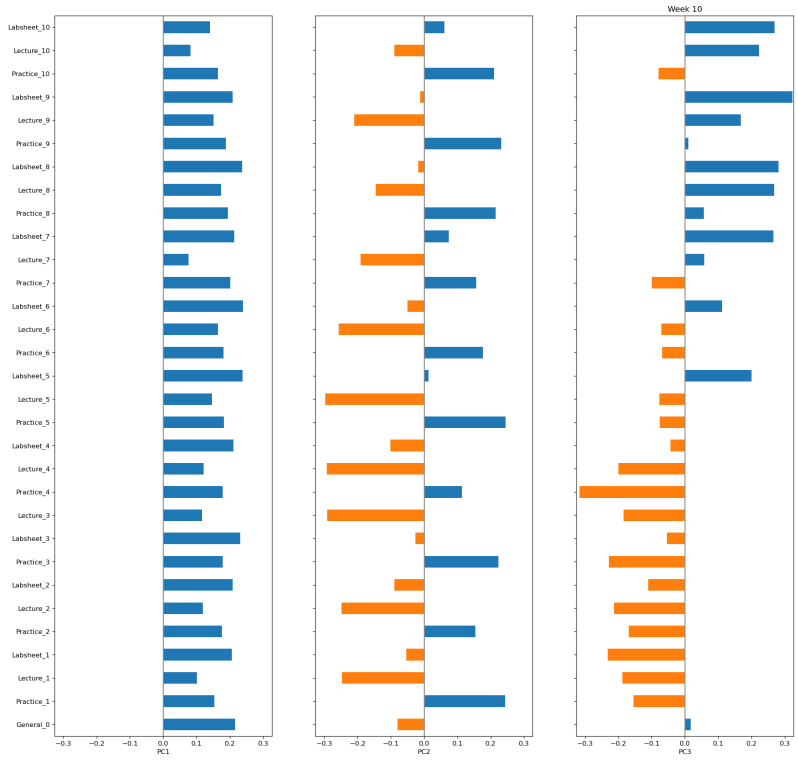
5.2.4 Student's Learning Behaviour during the COVID19 pandemic

During the COVID19 pandemic, students have experienced all their learning activities in an online setting. All lectures and interactions between students and

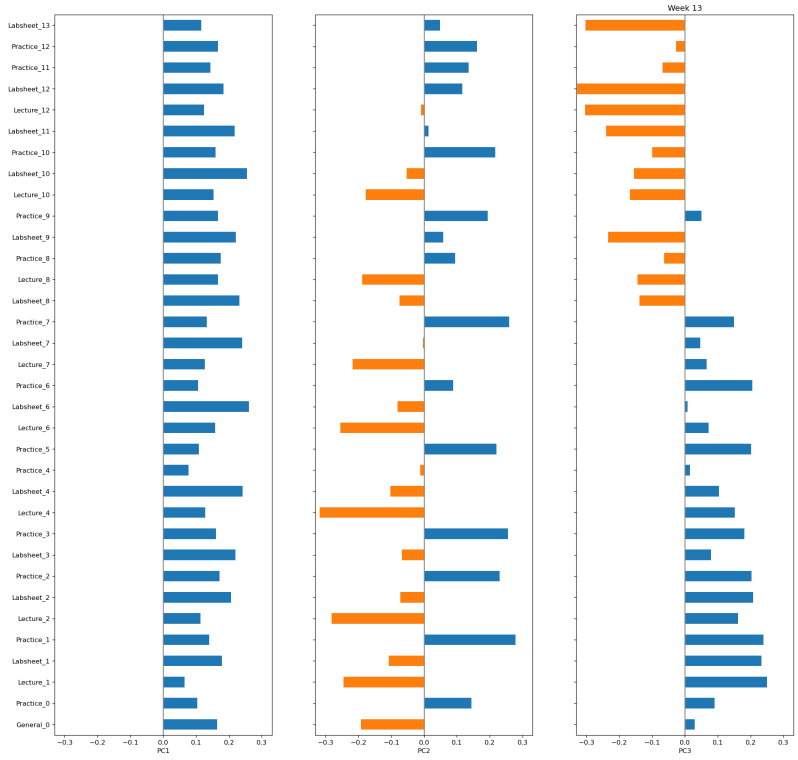
lecturers were conducted through an online meeting software and learning management systems and there was no any physical learning setting possible. With respect to the PCA analysis for the learning behavioural data collected during the COVID19 pandemic (i.e., Course#1-2020 and Course#2-2020), we also consider loadings and scores of the principal components for student cohorts. In Course#1-2020, the first three principal components were found to be significant based on RMT (see Section 5.2.2 and Figure 5.1c), which is similar to the data for 2018 and 2019. Regarding Course#2, unlike the pre-COVID19 data, Course#2-2020 also shows three significant principal components (see Figure 5.2c).

The loadings of selected principal components (i.e., PC1, PC2 and PC3) for Course#1 and Course#2 data are illustrated in Figure 5.7. The "trend effect" in the students' learning behaviours is also represented in the PC1 as all the loadings of the PC1 of both Course#1 and 2 in 2020 have positive values. This result indicates that all learning material items positively correlate with PC1. Hence, students during the COVID19 pandemic also tend to interact similarly with all given learning items during the semester.

Regarding PC2, in both Course#1-2020 and Course#2-2020, most of the loadings for Practice items have the opposite sign to the loadings for Lecture and Labsheet items. PC2 still reflects the "learning material item dimension", which is similar to the data before the pandemic. However, there is a slight difference in value sign of the PC2 loadings' between the data during and before the pandemic. In particular, in Course#1-2018 and Course#1-2019, the loadings of Practice and Labsheet are generally located on the same side (see Figure 5.5). However, most of the loadings of Practice items have opposite value signs with the Labsheet and Lecture items in the 2020 data. In other words, students appeared to have similar behaviours in using Lecture and Labsheet items during the lockdown while before the COVID19, the similarity between Labsheet and Practice has been shown. Possibly, when students had to study in a completely online setting, the chance for them to communicate with the instructors and other students while doing programming exercises is limited, so they mostly relied on reading lecture notes. On the other



(a) Course#1-2020

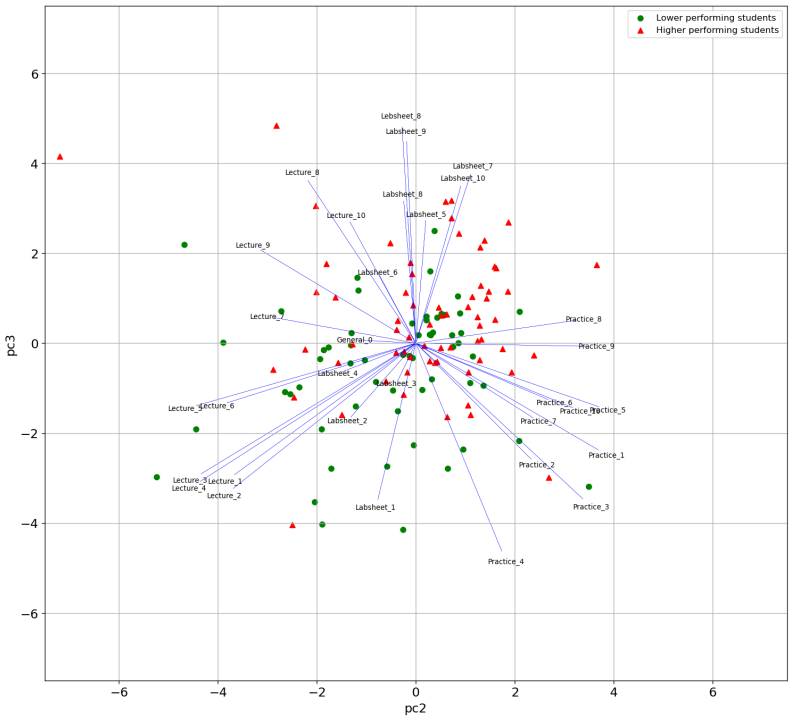


(b) Course#2-2020

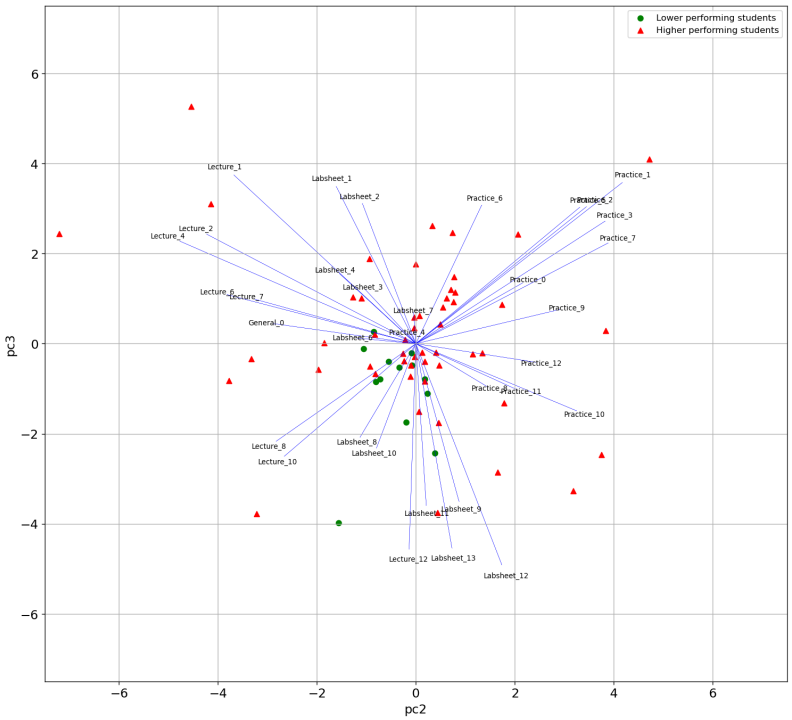
Figure 5.7: The loadings (eigenvector components) of the PC1, PC2 and PC3 extracted from the Course#1 and Course#2 datasets for the year of 2020 (during COVID19). The blues refer to positive values and the oranges refer to negative values

hand, the PC3 of Course#1-2020 and Course#2-2020 show a similar pattern to the PC3 of the pre-COVID19 data, reflecting the "time dimension" of the students' learning behaviours. There could be a deviation in students' interaction with the learning items at the beginning and end of the semester. In order to have more insights into the learning behaviours between student cohorts (i.e., higher and lower performing), we consider the role of the PC scores through biplots.

Figure 5.8 contains biplots of the PC2 and PC3 of Course#1-2020 (Figure 5.8a) and Course#2-2020 (Figure 5.8b), showing the combination between the scores and loadings of the principal components. In Course#1-2020 (Figure 5.8a), we can observe that most red dots are located on the left side of the graph, i.e., the same side with Practice items. Nevertheless, most green dots are plotted on the right side of the graph along with Lecture and Labsheet learning items. This observation shows evidence for the pattern that while higher performing students tend to focus more on working with given exercises, while lower performing students are likely to put more effort into reading lecture notes. Additionally, we notice that, when studying in the lockdown, a large number of lower performing students are observed to be more active in working with Labsheet items, which is not seen in the courses before the lockdown (see Section 5.2.3 and Figure 5.6). This behaviour may not be expected as typically students need to read instructions and requirements in labsheets before solving programming questions, which makes Practice and Labsheet items likely to go along with each other (as can be seen in the pattern for the pre-COVID19 datasets). In this context, it is possible that these lower performing students actually tried to do programming exercises at first, as they opened and navigated through labsheet items. However, these students might quit practising when facing challenging problems and not readily having someone to ask or communicate with during the lockdown. Some other students also might face other issues such as mental health or losing learning motivation to complete the tasks. This phenomenon appears to be more obvious in Course#2-2020 where almost all green dots, i.e., lower performing students, are located on the left side of the Figure 5.8b along with Labsheet and Lecture items.



(a) Course#1-2020



(b) Course#2-2020

Figure 5.8: The biplot of the PC2 and PC3 extracted from the datasets for Course#1 and Course#2 in year of 2020 (during the COVID19).

5.3 Community Detection for Students' Learning Behaviours using transition frequency features

In Section 5.2 above, we have conducted a PCA analysis with the support of RMT, and the patterns of learning behaviours of students in Course#1 and Course#2 have been revealed. However, the PCA above may contain limitations. The analysis focuses only on the top principal components and ignores the remaining components despite the fact that these components may still contain some information. In addition, the PCA analysis uses grade-based cohorts of students when in fact, many students with contradictory learning outcomes still show similar behaviours. Hence, we expect to find representative communities, i.e., groups of students, for higher and lower performing students. To this end, we follow the Community Detection approach (See Section 4.5). Furthermore, we also consider the sequential aspect of students' learning behaviours, i.e. the sequential order in the use of the learning items by using the *transition frequency features* (see Section 4.2.2). The details of the analysis are discussed below:

5.3.1 Data extraction and method

In this section, the *transition frequency features* from the event logs of Course#1 and Course#2 over the three academic years (ie. 2018, 2019 and 2020) have been extracted. The concept of *transition* refers to the phenomenon that a student switches from an action on a learning material item to the next action on the same or other learning material item when interacting with the learning system. For example, when the student s_1 scrolls down the page Lecture_1, then clicks on the link to open the page Labsheet_1, the following transitions can be recorded, i.e., *Lecture_1-Lecture_1* and *Lecture_1-Labsheet_1*. Further detail of the *transition frequency features* can be found in Section 4.2.2.

Six datasets are extracted from the event logs, namely Course#1-2018, Course#1-

Table 5.2: Detail of the datasets for Community Detection Analysis

Dataset	Number of students (columns)	Number of transitions (rows)	Number of Higher performing students	Number of Lower performing students
Course#1-2018	112	825	54	58
Course#1-2019	151	878	87	64
Course#1-2020	128	602	69	59
Course#2-2018	62	406	44	18
Course#2-2019	48	423	33	15
Course#2-2020	65	501	52	13

2019, Course#1-2020 and Course#2-2018, Course#2-2019 and Course#2-2020. Please note that these six datasets are extracted based on the *transition frequency features* and different to the datasets used in Section 5.2. A summary of the extracted datasets can be seen in Table 5.2.

For each dataset, we calculate its correlation matrix. Next, in order to remove the effect of noise and trend in the data, we make use of the method outlined in Section 4.4.1, i.e., cleaning the correlation matrix. Then, the cleaned correlation matrix is used to calculate a distance matrix of students' learning behaviours. A graph is constructed from the distance matrix using the Minimum Spanning Tree (MST) algorithm (see Section 4.5.3), which reflects the connection between a student and other students in terms of the similarity between their learning behaviours. In other words, for the dataset for each course, we construct a graph as an MST to display the similarity and dissimilarity of the students' learning behaviours. In the next step, community detection algorithms (i.e. the Girvan-Newman and Louvain algorithms) have been applied to the constructed graph, detecting communities of students who have the most similar learning behaviours. Students in each module can be divided into a smaller number of communities based on the distance between their learning behaviours and other learners' behaviours. For evaluation purposes, we also compare the ability to detect better communities between the cleaned and original datasets. Finally, we investigate the relationship between the detected communities and their learning outcomes.

Table 5.3: Community detection summary for Course#1. Groups are order in descending order based on the average grades of their members.

Group	Course#1-2018		Course#1-2019		Course#1-2020	
	Number of students	Average grade	Number of students	Average grade	Number of students	Average grade
Group 1	18	**0.79**	12	**0.89**	19	**0.71**
Group 2	11	0.52	16	0.64	18	0.59
Group 3	15	0.5	21	0.61	19	0.56
Group 4	17	0.42	25	0.57	15	0.43
Group 5	11	0.25	17	0.32	19	0.42
Group 6	13	0.21	14	0.32	11	0.38
Group 7	13	0.17	20	0.31	15	0.36
Group 8	14	**0.05**	26	**0.25**	12	**0.08**

5.3.2 Selecting community structure

We note that the number of communities to be detected by the Girvan-Newman algorithm can be configurable depending on analysis purposes, forming a community structure. In this research, we use the concept of *mixed community rates*, i.e. a good community structure should contain fewer *mixed community rate* and higher number of *higher* or *lower performing communities*. Details of the *mixed community rates* are described in Section 4.5.4.

Figure 5.9 and Figure 5.10 show the investigation of *mixed community rate* for each possible community structure detected by the algorithm in Course#1 and Course#2 over the three academic years. Indeed, the number of detected communities can go up to the total number of students in the whole graph. However, we do not want a fragmented community structure where each community contains only a few students. Hence, we merely show a part of the possible community structures in both figures.

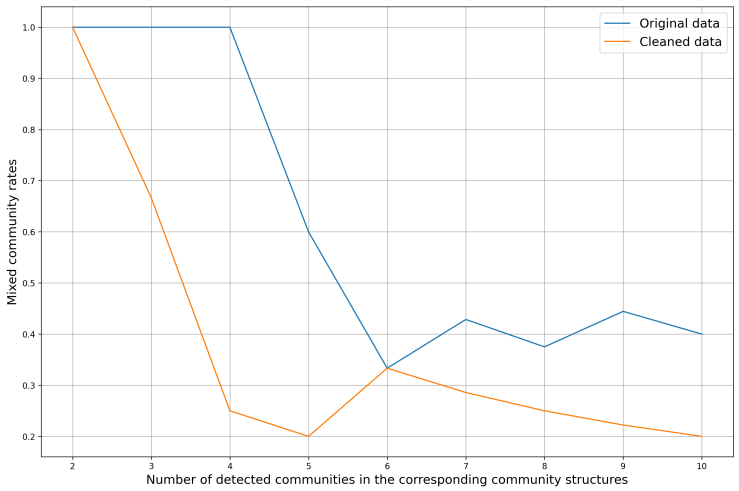
Both Figures 5.9 and 5.10 show that the cleaned dataset has a better support for community detection in comparison with the original dataset with the lower *mixed community rates*. Overall, the mixed community rate in the community structures detected using the cleaned datasets is lower than the figures for the original datasets for Course#1. We also observed a similar phenomenon for the Course#2 datasets,

Table 5.4: Community detection summary for Course#2.

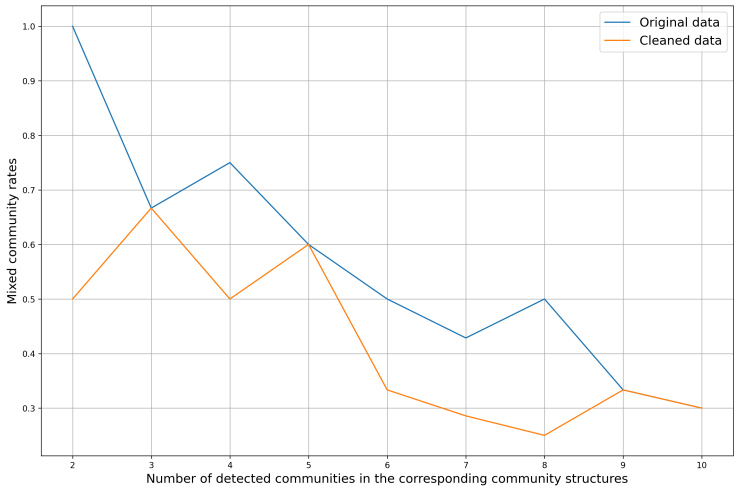
Group	Course#2-2018		Course#2-2019		Course#2-2020	
	Number of students	Average grade	Number of students	Average grade	Number of students	Average grade
Group 1	05	**0.75**	05	**0.95**	08	**0.84**
Group 2	05	0.60	07	0.92	09	0.80
Group 3	05	0.60	05	0.80	10	0.77
Group 4	09	0.58	06	0.79	08	0.71
Group 5	09	0.44	06	0.66	08	0.65
Group 6	04	0.43	05	0.65	08	0.59
Group 7	04	0.43	05	0.60	08	0.59
Group 8	08	0.43	09	**0.55**	06	**0.50**
Group 9	07	0.39	NA	NA	NA	NA
Group 10	05	**0.33**	NA	NA	NA	NA

although the gap seems to be not as clear as Course#1. This may be due to the fact that Course#2 is generally less challenging in terms of the amount of knowledge and the level of difficulty in exercises. Hence, the difference in learning behaviours among students can be blurred.

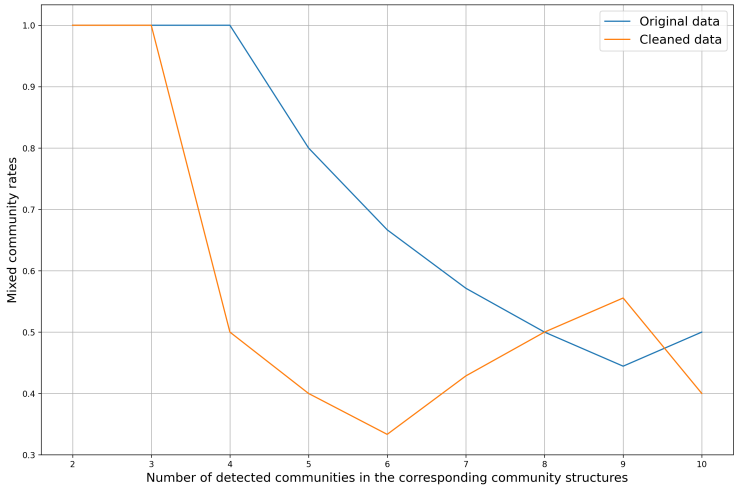
In addition, based on these figures, it is possible to determine the size of the community structures. In particular, we can choose an option among the low points of the mixed community rate line to determine how many communities are to be detected, subject to analysis purposes. The detected results can be seen in Table 5.3 for Course#1 and Table 5.4 for Course#2. In Table 5.3, for each year, eight groups have been detected with the number of students in each group and the average grades of all students in the group for the final lab exam in week 12. Similarly, Table 5.4 displays ten detected groups for Course#2-2018 and eight groups for Course#2-2019 and Course#2-2020. All groups are ordered from the highest to the lowest average grades in the tables. We notice from the detected communities that some groups mostly include higher performing students (based on their grades), while other groups also mostly contain lower performing students. Further visualisation of the detected results can be seen as MST graphs in Appendix A and dendrograms in Appendix B. Given those students in a detected group/community have similar learning behaviours, we conclude that there is a relationship between students'



(a) Course#1-2018

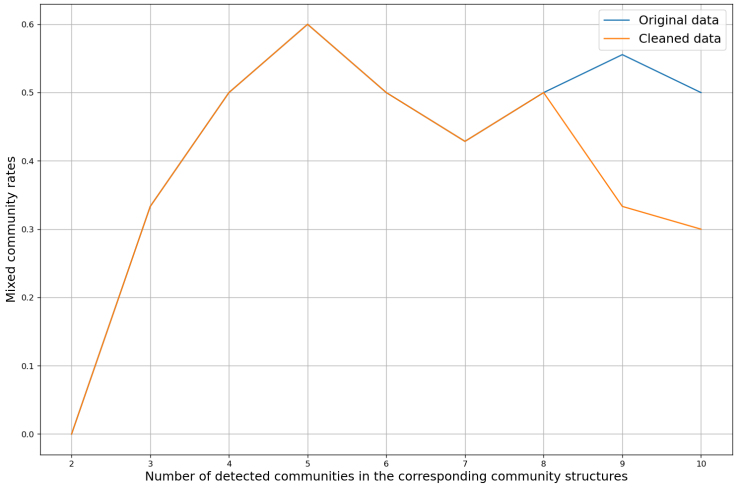


(b) Course#1-2019

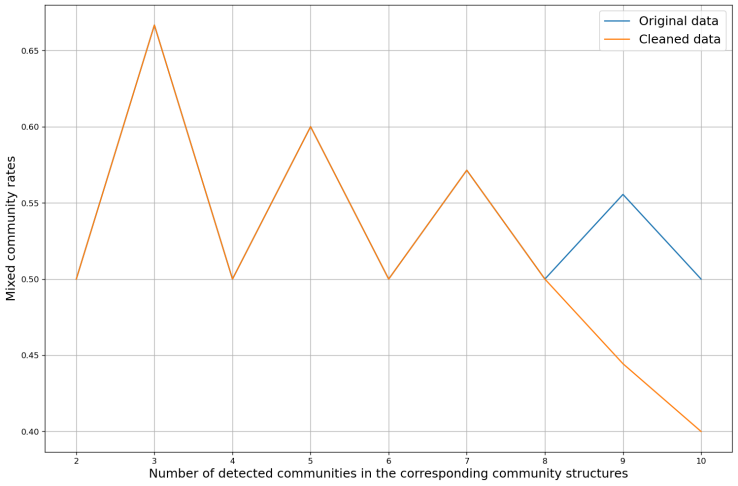


(c) Course#1-2020

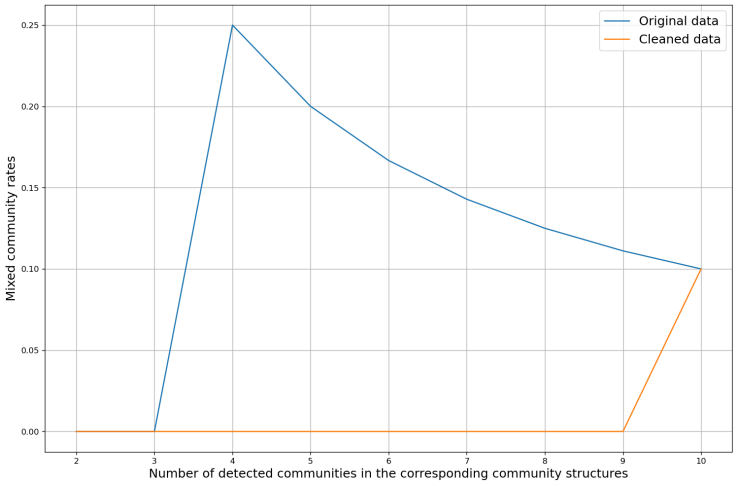
Figure 5.9: Mixed community rates in community structures for Course#1 over three academic years.



(a) Course#2-2018



(b) Course#2-2019



(c) Course#2-2020

Figure 5.10: Mixed community rates in community structures for Course#2 over three academic years.

learning behaviours and learning outcomes. Once the communities are detected, we can focus more on representative communities to figure out the study pattern of students. For example, we can pick up the two communities with the highest and lowest average grades to see if the students' learning behaviours in those groups are different. The analysis results of this is described in the next subsection.

5.3.3 Difference between the highest vs lowest performing communities

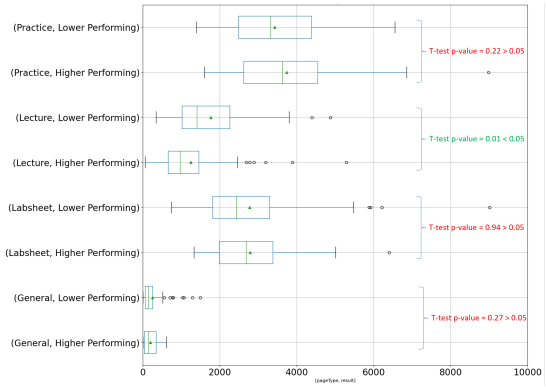
The higher and lower performing communities in Tables 5.3 and 5.4 can be selected for further investigation of the difference in interactions with course material items among the student cohorts. For example, we combined the top four groups in Table 5.3 (i.e., Group 1,2,3,4 with higher average grades) to form a higher performing detected community in Course#1 and the bottom four groups (i.e, Group 5,6,7,8 with lower average grades) to form a lower performing detected community. The similar action has been done for Course#2. These detected communities have been used in the comparison with the grade-based cohorts in terms of detecting the difference of learning behaviours between higher and lower performing student cohorts.

The comparative results between the higher and lower performing student cohorts of Course#1 are presented in Figure 5.11 while Figure 5.12 shows the results for Course#2. Particularly, each figure contains a set of subfigures that represent the number of interactions on different types of learning material items between higher and lower performing student cohorts. Please note that the subfigures on the left side represent the result for grade-based cohorts in which students are classified into higher/lower performing cohorts based on their grades. These figures are, therefore, similar to the figures discussed in Chapter 3 - Section 3.3 which gives an exploratory analysis of the collected data. On the other hand, the subfigures on the right side of Figures 5.11 and 5.12 illustrate the data for the communities with the higher and lower performing detected communities for Course#1 and Course#2.

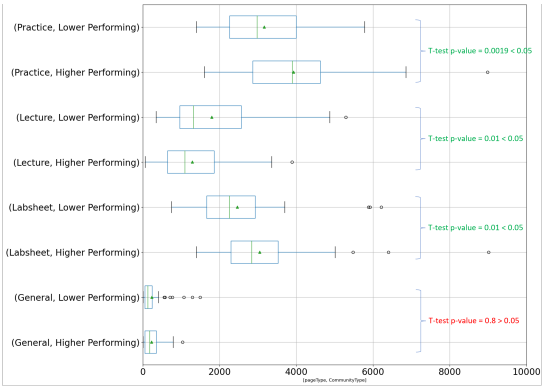
In general, we notice that the detected communities appear to show a more

distinct difference between the higher/lower performing students than the figures for grade-based cohorts. For example, in Figure 5.11a - using grade-based cohorts - while the higher performing students tend to have a higher number of learning events on Practice items than lower performing students, the two boxplots seem to "overlap" with the T-test p-value = $0.22 > 0.05$. This indicates that much lower performing students still have a number of practice activities that are comparable with higher performing students. This phenomenon may refer to the effect of the noise and trend in the datasets. Nevertheless, the study pattern appears to be more clear in Figure 5.11b. Higher performing communities have a significantly higher number of practice events than lower performing communities (T-test p-value = $0.0019 < 0.05$). This observation is consistent with the PCA analysis result reported in Section 5.2, which claims that higher performing students are likely to be more active in doing programming exercises. A similar pattern is also noticeable to the use of Labsheet items. The results in Figures 5.11d and 5.11f also show a similar finding.

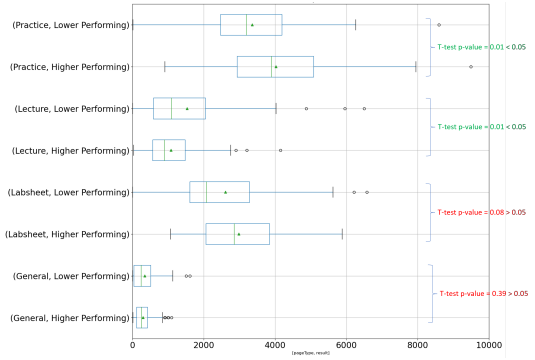
Regarding Lecture items, in Course#1, the result for the pre-COVID19 datasets shows that lower performing students tend to be far more active in reading lecture notes than higher performing students. Nevertheless, during the COVID19 pandemic, there is no difference between the two communities in using Lecture items. This observation is also consistent with the finding in PCA analysis, that when studying from home under lockdown, students might have limited opportunities to interact with the instructors and other classmates. Hence, they tend to merely rely on the given online learning materials. A similar phenomenon has been also seen in Course#2 where higher and lower performing communities show the same level of interaction with Lecture items during the COVID19 pandemic. However, even before the lockdown, Cohort#2 students also appear to show not much difference in using Lecture items, which was not witnessed in Course#1. This could be due to the fact that Business Informatics students in Course#2 might not have solid pre-requisite and pre-knowledge for studying programming in comparison with Software Engineering students in Course#1. Hence, Course#2 students might need to rely



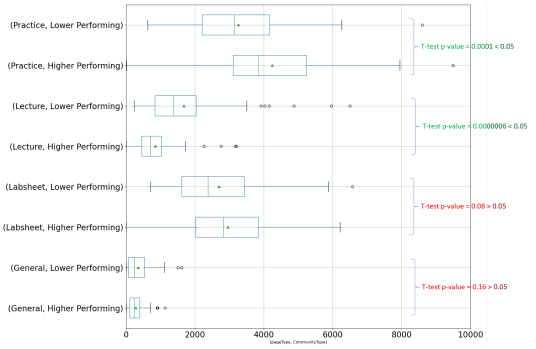
(a) Course#1-2018 - grade-based cohort



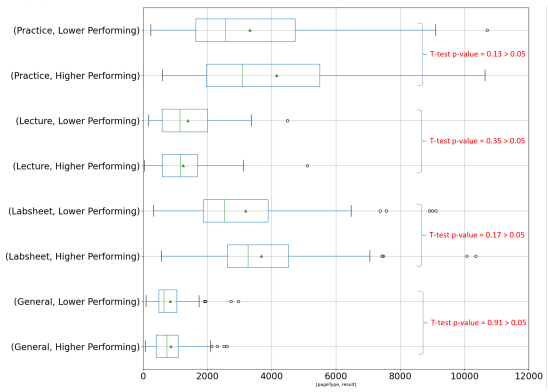
(b) Course#1-2018 - detected community



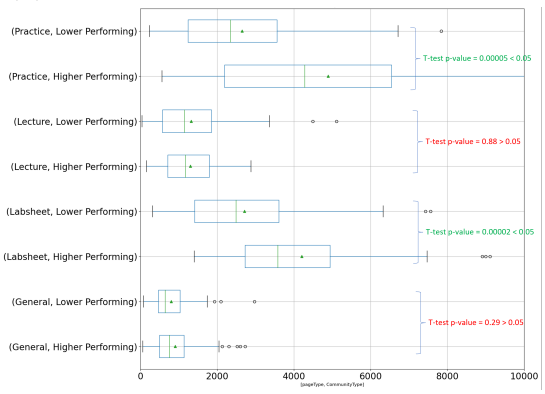
(c) Course#1-2019 - grade-based cohort



(d) Course#1-2019 - detected community

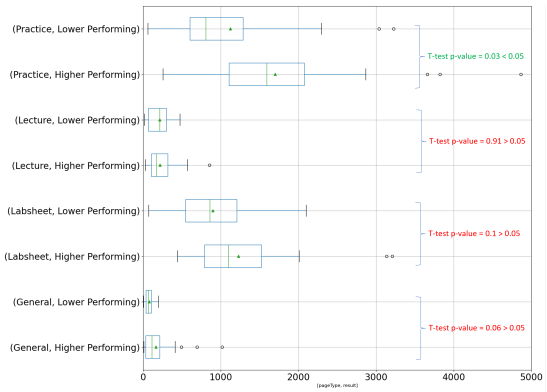


(e) Course#1-2020 - grade-based cohort

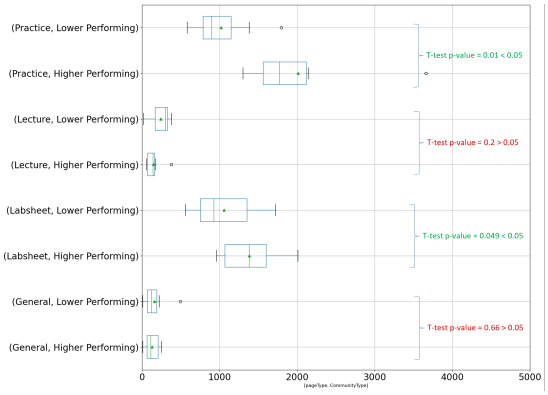


(f) Course#1-2020 - detected community

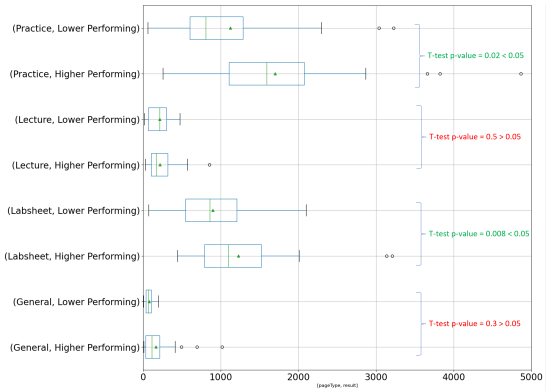
Figure 5.11: Learning behaviours of detected communities vs grade-based cohorts for Course#1 over three academic years.



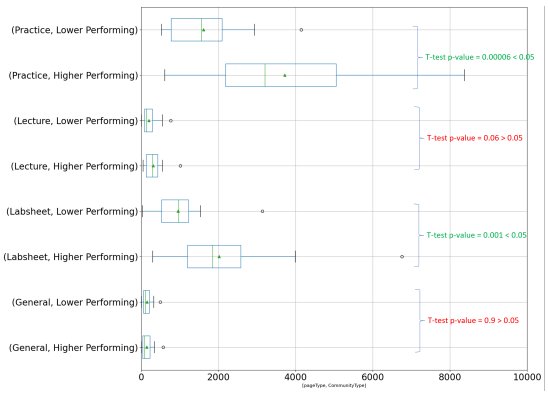
(a) Course#2-2018 - grade-based cohort



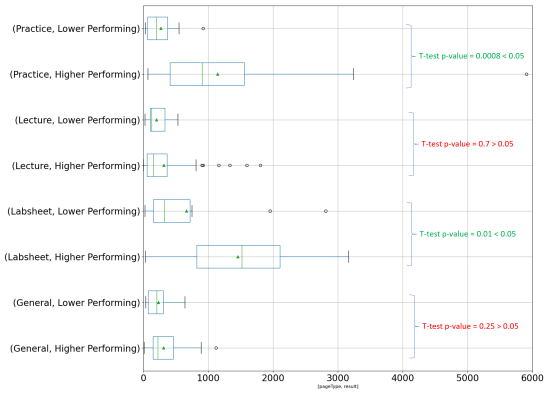
(b) Course#2-2018 - detected community



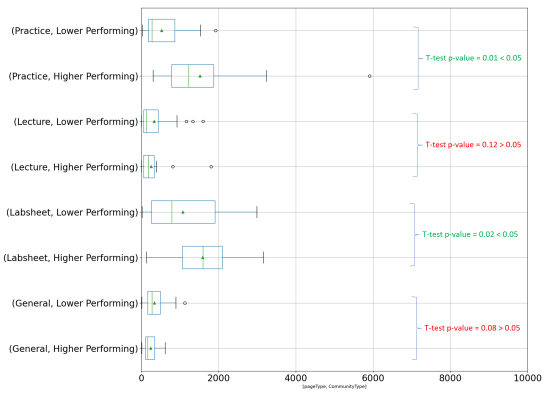
(c) Course#2-2019 - grade-based cohort



(d) Course#2-2019 - detected community



(e) Course#2-2020 - grade-based cohort



(f) Course#2-2020 - detected community

Figure 5.12: Learning behaviours of detected communities vs grade-based cohorts for Course#2 over three academic years.

more on lecture notes.

On the other hand, in all cases, there is no difference in using General items, i.e., the documents that contain general information about the courses such as the timetable, marking scheme etc., between student cohorts over the years. This is understandable as almost all students may need to visit those documents a few times during the study period, and thus the behaviours on those learning items generally could not explain the learning performance of students.

We can go further with the analysis between the two communities by having a close look at the students' interaction for each learning item each week. For example, Table 5.5 demonstrates the difference in the number of learning activities in using learning material items in Course#1-2018 and Course#1-2019 between the two communities, i.e., higher and lower performing, while the results for the Course#2-2018 and Course#2-2019 can be seen in Table 5.6. For each item, non-parametric statistical tests (i.e. Mann-Whitney U Test [117]) have been used to verify if there is a significant difference between the highest and lowest performing communities in terms of using the item during the courses. The course material items in which the highest and the lowest performing communities have a significant difference in the number of events (p-value < 0.05) are highlighted.

Table 5.5: Highest vs Lowest performing communities in Course#1 for 2018 and 2019. The cell values refer to the average number of interactions with learning items in a community. The asterisks indicate the learning items where there is a significant difference between the two communities ($p - value < 0.05$). In particular, * if there is only a significant difference in Course#1-2018 only, ** if there is only a significant difference in Course#1-2019 only and *** if there are significant differences in both academic years.

Items	Course#1-2018		Course#1-2019	
	Highest performing community	Lowest performing community	Highest performing community	Lowest performing community
General	160.0	159.0	403.0	355.0

Continued on next page

Items	Course#1-2018		Course#1-2019	
	Highest performing community	Lowest performing community	Highest performing community	Lowest performing community
Lecture1**	43.0	56.0	57.0	**70.0**
Lecture2*	53.0	**129.0**	121.0	110.0
Lecture3*	15.0	**157.0**	129.0	112.0
Lecture4*	33.0	**72.0**	50.0	47.0
Lecture5***	53.0	**98.0**	76.0	**79.0**
Lecture6***	36.0	**122.0**	148.0	**150.0**
Lecture7***	31.0	**87.0**	82.0	**104.0**
Lecture8**	32.0	38.0	48.0	**59.0**
Lecture9**	128.0	167.0	**140.0**	134.0
Lecture10*	91.0	**102.0**	91.0	76.0
Lecture11***	15.0	**42.0**	81.0	**138.0**
Lecture12**	25.0	2.0	61.0	**106.0**
Labsheet1***	167.0	**245.0**	**164.0**	66.0
Labsheet2**	283.0	352.0	**440.0**	235.0
Labsheet3***	199.0	**263.0**	**313.0**	116.0
Labsheet4	95.0	100.0	206.0	121.0
Labsheet5**	325.0	223.0	**442.0**	208.0
Labsheet6	244.0	213.0	410.0	225.0
Labsheet7***	**245.0**	139.0	**384.0**	194.0
Labsheet8	85.0	85.0	162.0	107.0
Labsheet9**	204.0	132.0	**451.0**	194.0
Labsheet10**	256.0	140.0	**345.0**	137.0
Labsheet11***	**163.0**	100.0	**159.0**	104.0
Labsheet12*	**80.0**	27.0	120.0	133.0

Continued on next page

Items	Course#1-2018		Course#1-2019	
	Highest performing community	Lowest performing community	Highest performing community	Lowest performing community
Practice1	286.0	379.0	223.0	52.0
Practice2***	427.0	**613.0**	**417.0**	190.0
Practice3	266.0	348.0	355.0	168.0
Practice4*	173.0	**308.0**	387.0	222.0
Practice5**	254.0	185.0	**382.0**	217.0
Practice6***	**277.0**	185.0	**346.0**	206.0
Practice7***	**246.0**	88.0	**490.0**	390.0
Practice8**	251.0	263.0	**607.0**	232.0
Practice9***	**360.0**	138.0	**493.0**	215.0
Practice10***	**291.0**	89.0	**492.0**	315.0
Practice11***	**328.0**	102.0	**502.0**	426.0
Practice12***	**234.0**	163.0	**639.0**	367.0

Table 5.6: Highest vs Lowest performing communities in Course#2. The cell values refer to the average number of interactions with learning items in a community. Statistical tests were not conducted for this result because there are merely a small number of students in the two communities(i.e. 5 and 6 students).

Items	Course#2-2018		Course#2-2019	
	Highest performing community	Lowest performing community	Highest performing community	Lowest performing community
General	211.0	231.0	93.0	107.0
Lecture1	37.0	100.0	93.0	19.0
Lecture3	54.0	86.0	32.0	30.0
Lecture5	42.0	43.0	29.0	31.0

Continued on next page

Items	Course#2-2018		Course#2-2019	
	Highest performing community	Lowest performing community	Highest performing community	Lowest performing community
Lecture9	65.0	19.0	59.0	5.0
Labsheet1	168.0	128.0	269.0	163.0
Labsheet2	127.0	56.0	228.0	98.0
Labsheet3	183.0	68.0	256.0	104.0
Labsheet4	203.0	68.0	315.0	123.0
Labsheet5	169.0	60.0	227.0	60.0
Labsheet7	247.0	144.0	151.0	59.0
Labsheet8	196.0	63.0	104.0	8.0
Labsheet9	29.0	9.0	203.0	18.0
Labsheet10	91.0	15.0	145.0	29.0
Labsheet11	40.0	4.0	96.0	12.0
Practice1	54.0	20.0	112.0	60.0
Practice2	115.0	52.0	335.0	35.0
Practice3	180.0	22.0	452.0	43.0
Practice4	171.0	24.0	400.0	57.0
Practice5	129.0	138.0	453.0	281.0
Practice6	109.0	66.0	161.0	31.0
Practice7	675.0	94.0	500.0	1.0
Practice8	247.0	26.0	382.0	20.0
Practice9	67.0	22.0	345.0	4.0
Practice10	81.0	0.0	427.0	0.0
Practice11	84.0	65.0	274.0	440.0

In addition, as seen both in both Tables 5.5 and 5.6, students in the highest performing community appeared to be more active than the lowest-performing community, with a higher average number of learning events in all Practice and Labsheet

items. These gaps are likely to increase over time during the semester of the both courses in 2018 and 2019. For example, in Course#1-2018, the average number of events in *Practice_11* (i.e. practice items in week 11) of the highest performing community is 328, which is about three times higher than the result for the lowest performing community (102). A similar phenomenon can be observed in the data of other cohorts. Nevertheless, students in the lowest performing community are recorded to create a higher number of events in Lecture items for both classes of Course#1 and Course#2-2018. For example, the number of events in lecture notes in weeks 2-7 created by the lowest performing community is about two to three times higher than the figures for the highest performing community in Course#1.

5.3.4 Girvan Newman vs Louvain methods

We compare the Girvan Newman community detection results selected above (Table 5.3 and 5.4) and the corresponding results produced by the Louvain method in the six datasets of the two courses (Course#1 and #2) over the three academic years. Table 5.8 illustrates an investigation of the possible difference between the Louvain and Girvan Newman method when both algorithms are applied to the cleaned data and original data for the four datasets, using v-measure (see Section 4.5.4). Overall, the results from the Louvain and Girvan Newman method appear to be broadly similar. There are strong agreements between the community detection results of both algorithms with v-measure scores being greater than or equal to 0.82 in all cases with either cleaned or original data. It can also be seen that the v-measure scores for the cleaned data tend to be higher than those of the original data. It is possible that the proposed cleaning method could support the reduction of variation between the two algorithms when they are applied to the same dataset. However, we note that this finding needs to be verified with more datasets.

Additionally, the results in Table 5.7 indicate the values for *mixed community rates* of the community structure detected by the Louvain method. The rates for the cleaned data are likely to be lower than those for the original data. This is consistent

Table 5.7: Comparison of mixed community rate of community structure produced by Louvain method between Cleaned and Original data. The cell values refer to the mixed community rate of the community structure for each orginal and cleaned dataset.

Dataset	Cleaned data	Original data
Course#1-2018	0.25	0.55
Course#1-2019	0.33	0.42
Course#1-2020	0.4	0.6
Course#2-2018	0.25	0.5
Course#2-2019	0.12	0.43
Course#2-2020	0.0	0.28

Table 5.8: v-measure score of the clustering results detected by the Louvain and Girvan-Newman algorithms for Cleaned and Original data.

Dataset	Cleaned data	Original data
Course#1-2018	0.86	0.86
Course#1-2019	0.84	0.82
Course#1-2020	0.85	0.84
Course#2-2018	0.84	0.82
Course#2-2019	0.87	0.89
Course#2-2020	0.82	0.82

with the application of the Girvan-Newman method, i.e. the cleaned data can also support the Louvain method to deliver better community detection results with the lower number of mixed communities.

5.4 Learning behaviours and outcome prediction

In this section, we conducted experiments to verify the ability to predict students' learning outcomes using learning behavioural features. This could bring an opportunity to build a classifier to identify "at-risk" students, for this classifier we only need data automatically collected from the learning management system during the study period rather than historical data.

5.4.1 Data and method

To evaluate the predictability of the students' interaction with course material items for learning outcomes, we use the *student-event item data matrix* as the input variable. In particular, we combine the *student-event item data matrix* of the datasets for Course#1-2018 and Course#1-2019 into a single tabular dataset. The reason for this selection and combination is that Course#1 in the 2018 and 2019 had similar setting in terms of the instructor, lecture material items and timetable. Meanwhile, the Course#1-2020 was delivered in 10 weeks instead of 12 weeks. Therefore, the weekly lectures had to be changed accordingly. In addition, the dataset for Course#1 in both years of 2018 and 2019 contain greater sample size with 263 students in total, which is far greater than Course#2 with 110 students (see Table 5.1).

Next, we conduct the Cleaning the Dataset method, forming *fully cleaned data* and *partly cleaned data*. In particular, for *fully cleaned data*, the values of α and β are all set to 1, i.e., the components for noise and trend are completely removed. On the other hand, for the construction of *partly cleaned data*, we set $\alpha = 0.5$ and $\beta = 0.8$ with the assumption that the noise and trend components still contain some useful information for the building of the prediction models. The detail about cleaning the Dataset method has been described in Section 4.4.2.

For comparison purposes, we also use the original and PCA-transformed datasets as predictors. In the original dataset, no pre-processing actions have been conducted. With respect to PCA-transformed datasets, we select a number of principal components such that they can explain a relatively large proportion of variance in the datasets, i.e. 70% and 80%, labelled as *PCA at 70% variance* and PCA at 80% variance, respectively.

The target variable is defined based on the student's scores on each formative assessment. There are three formative assessments for Course#1 on Week 4, 8 and 12. As stated above, we classify students who achieved more than 40% in a lab exam as *higher-performing* students and the remaining as *lower-performing* students.

For each week, a *student-event data matrix* has been extracted from the corre-

sponding weekly event log. The data collected in a certain week contain recorded learning events from the beginning of the course to that week. Then, the weekly data was used to predict the student results for the next assessment. For example, in Course#1, the data collected in weeks 1, 2, 3, 4 are used to predict the results of the lab exam 1; the data in weeks 5, 6, 7 and 8 are used for predicting lab exam 2 results and the remaining weekly data are used to forecast the last exam result.

Each dataset serves as input data for all algorithms with the same parameter configuration in each technique. In addition, since the main target of this experiment is to evaluate the performance of classification algorithms on the pre-processed datasets, the 10-fold cross-validation technique has been applied [84]. We also use *ROC_AUC*, *Accuracy*, and *F1 scores*, to evaluate the prediction performance of each model.

5.4.2 Comparison of Dataset pre-processing strategies

Figures 5.14, 5.15 and 5.13 demonstrate the *ROC_AUC*, *Accuracy*, and *F1* scores from different models and input datasets, respectively. It can be seen that the three figures illustrate a similar pattern in the difference of the evaluation metrics among the models. Overall, fully and partly cleaned datasets appear to have better-predicting performances in comparison with other data preparation strategies. In particular, regarding the partly cleaned dataset, the Gradient Boosting outperforms other models in all three metrics (i.e. Accuracy: 0.79, F1 score: 0.75 and ROC_AUC: 0.81), along with XGBoost. In terms of the fully cleaned dataset, the models have shown that they have been well-performing in Gradient Boosting, KNN, Logistic Regression and SVM algorithms with relatively high scores in all the three metrics.

Conversely, models using the original and PCA datasets appear to have lower performances across all predicting algorithms with the scores roughly around 0.60 and 0.70. Meanwhile, although the PCA dataset at 80% variance has the lowest performance in Gradient Boosting and KNN, the dataset has shown its predictability with the Logistic Regression algorithm with the highest accuracy and f1 scores. It

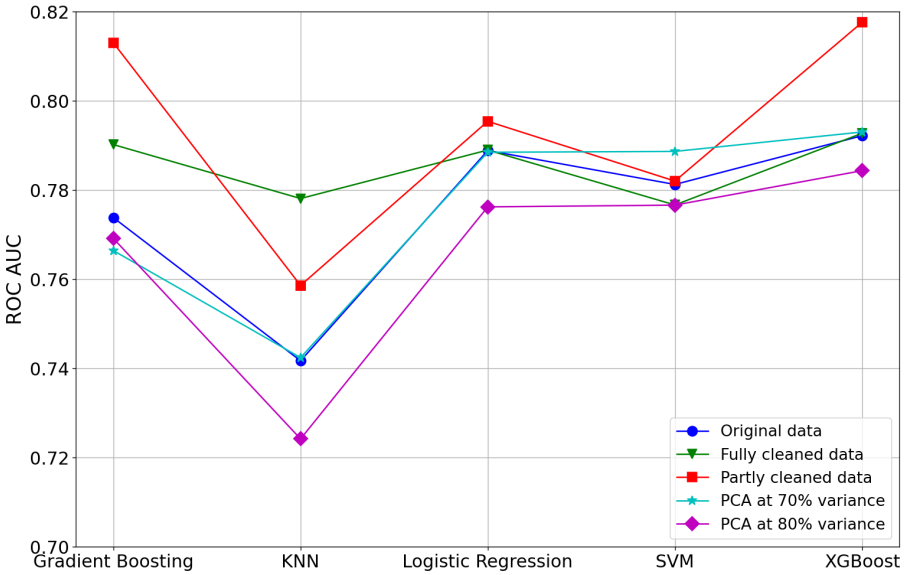


Figure 5.13: Comparison of the ROC_AUC scores of predicting models using different data pre-processing strategies.

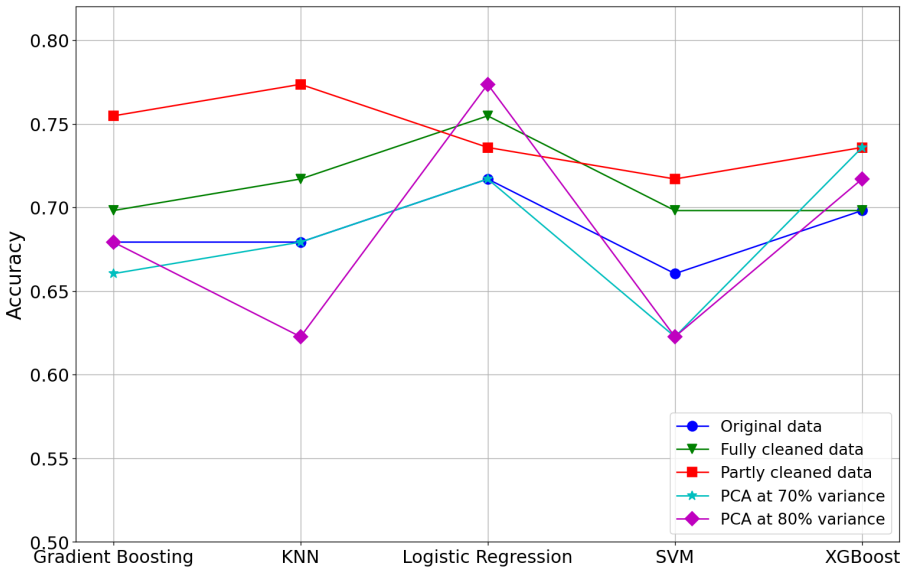


Figure 5.14: Comparison of the accuracy scores of predicting models using different data pre-processing strategies.

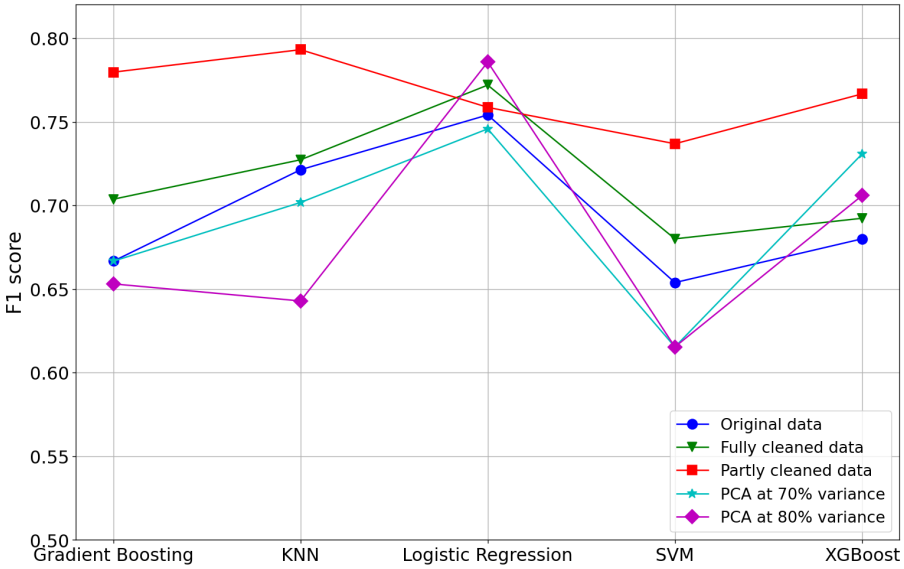


Figure 5.15: Comparison of the f1 scores of predicting models using different data pre-processing strategies.

is possible that when only top principal components were kept in the dataset, the noise part has been eliminated.

5.4.3 Early prediction investigation

Figure 5.16 illustrates the mean of the cross-validation on ROC_AUC score of the models using fully cleaned dataset over 12 weeks during the course while Figure 5.17 shows the results for the partly cleaned dataset. In general, most of the models can produce good predictions for the datasets after week 4. The ability of the models produced to classify students in the first four weeks is relatively poor, which is to be expected, probably due to the imbalance of the number of passed and failed students in lab exam 1. In fact, lab exam 1 usually comprises the easiest tasks which merely require the understanding of simple concepts in programming, e.g. using variables, operators and inputs. As a result, the majority of students usually pass the first exam. However, the difficulty level increases over lab exams 2 and 3, causing the target variable to become more balanced. Hence, the models can better predict the pass or failure of a student in lab exams 2 and 3.

Although the performance of the models increases over time with the growth of

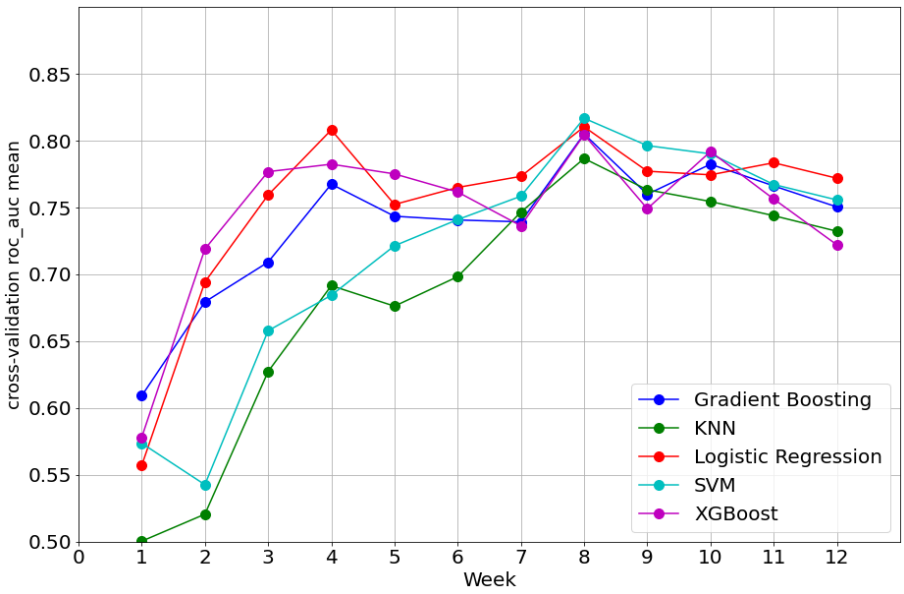


Figure 5.16: The 10-fold cross validation on ROC_AUC score of the models with fully cleaned data.

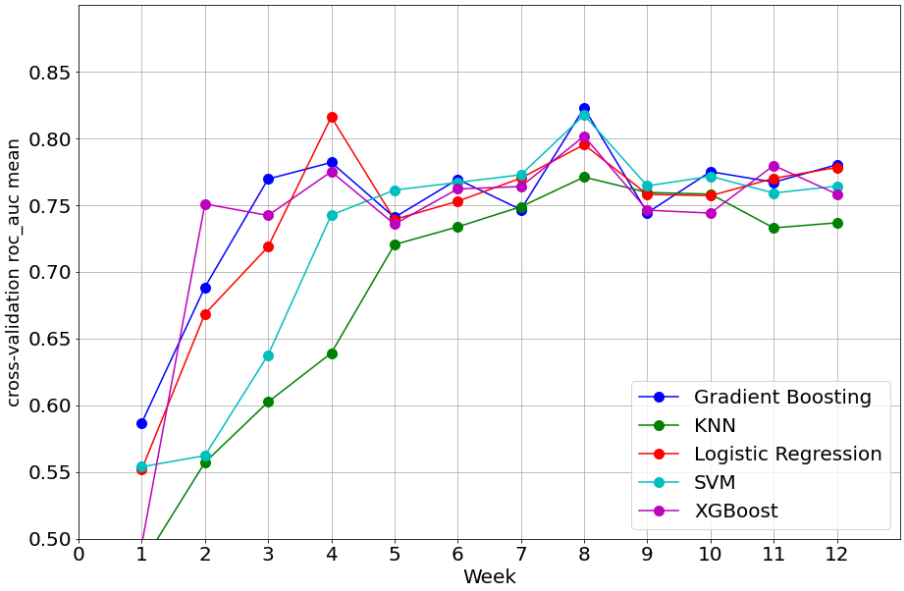


Figure 5.17: The 10-fold cross validation on ROC_AUC score of the models with partly cleaned data.

the data collected, early data can support relatively good prediction. For example, in Figure 5.16 the XGBoost model for week 5 data, which predicts the students' results of lab exam 2, achieves the ROC_AUC score of 0.78. The SVM model in week 9, which predicts the final exam in week 12, also achieves the acceptable result with a score of 0.80. While it is difficult to compare the two any more than qualitatively, we can say that our models seem to produce comparable if not better performance performance than a recent prediction model in a similar computing educational context [64] where the author achieved the ROC_AUC score of 0.73 with the SVM model for the prediction of the student learning outcome in the Data Structure course. Therefore, early learning behavioural data may contain signs of students' learning outcomes [116] and can be good predictors, holding the potential to be a "leading indicator" of "at-risk" students.

5.5 Conclusion

Until this point, it has been shown that the experimental results are able to answer all proposed research questions. The students' learning behaviours can be captured in the form of event log data and the behavioural features can be extracted to serve further analysis. In addition, it is possible to argue that there is a connection between the learning outcomes of students and their learning behaviours. Differences between the behaviours of higher and lower performing student cohorts have been observed. Of equal importance, the behavioural data acquired from the system can be used as a leading indicator in the prediction of students' success and failure in programming study.

Chapter 6 will give a further discussion about the impact of the results of this thesis on the wider context of the programming education and learning analytics methods.

Chapter 6

Discussion and Conclusions

6.1 Introduction

It is always important to emphasise the impact of research results in either an academic or a practical context. We would hope that this research can contribute to the improvement of the pedagogical value of programming education in higher education institutions. At the same time, our novel approaches in using a real students' learning behaviours are expected to pave the way for more insightful research being implemented in the future. These impacts are discussed in this chapter, along with the limitations of our research within the scope of this thesis.

6.2 Implications: Revisiting research questions

In this thesis, we have shown that students' learning behaviours can be captured and stored as learning event logs on a learning management system. The learning behaviour features can be extracted in the form of a *student-event data matrix* and a *transition-student data matrix*. We also propose a novel method to process the data, i.e., to remove the effects of noise and trend in the students' learning behavioural datasets. The benefits of the cleaned data have shown with better results delivered in the implementation of the Community Detection and learning outcome prediction in the context of Programming Education.

Based on the experimental results, we also believe that it is possible to say that there is a relationship between students' learning behaviours and their learning outcomes. We have found evidence of distinct communities of students in the dataset and that students who are grouped in the same community were likely to achieve similar exam results. In other words, students having similar learning behaviours tend to perform similarly in the exam. This finding is in agreement with [175, 96] where the authors have defined and analysed various learning behaviour styles with different learners' behaviours in perceiving and responding to learning environments. This is because the learning styles may have an effect on students' satisfaction and can also be a useful indicator of learning success [162].

Overall, the learning behaviours of students in Course#1 and Course#2 in both academic years tend to be similar from one year to the next. In both modules, we have found what seem to be differences between lower and higher performing communities. In particular, higher performing students were found to be more active in practising related items such as navigating lab sheets and doing exercise. In addition, the higher performing students consistently interacted with course material items and exercises during the courses. The lower performing students, however, appeared to lose their focus and motivation to practice, i.e. to actually to do programming practice exercises, towards the later stages of the study. This result is consistent with the initial investigation of programming [105] that *practice* is essential for improving students' programming skills. These findings, available so early in the semester, are essential for such core courses especially since it has been found that students should be given opportunities to practice and receive constructive feedback [21]. In [171], the authors indicated that programming skills may be improved if students practice frequently with the support from an interactive web-based environment for teaching programming. However, in the context of this research, students from the lower performing group might face challenges during their study progress at the later stage of the module, e.g. as the knowledge was becoming more difficult to understand. As a consequence, they might lose their confidence and motivation to actively participate in practical sessions, additionally highlighting the need for

early intervention and encouragement. This could be a reason why lower-performing students seem to be more active in reading lecture notes than higher performing students, as observed in the research. This could be due to the fact that, along with course books, higher-performing students tend to use alternative references from external sources such as consulting senior students and tutorials from the Internet to support their understanding [138]. It is possible to conclude that lower-performing students might face difficulties in understanding new concepts, and, as a result, they merely keep reading lecture notes while becoming discouraged and unmotivated to study and practice, especially in the later phase of the semester when the knowledge becomes more difficult to acquire effectively.

Additionally, we noticed that the lower performing students tend only to attempt their programming tasks according conventional common methods rather than creatively trying different approaches based our observation. As a result, they usually upload solutions once and move to other tasks. In contrast, the higher performing students tend to try various approaches for a given programming task and then usually submit them all at once, leading to a higher number of events in practical items logged on the system, in comparison with the practical activities of lower performing communities.

In both Course#1 and Course#2, higher performing students are more likely to be more active on Practice items, i.e., solving programming tasks. However, we also found that there seems to be a distinction between the learning behaviours of the Course#1 and Course#2 cohorts. In Course#1, the lower performing students appear to focus more on reading lecture notes than higher performing students. However, this phenomenon is not observed in Course#2. In particular, in Course#2, there is almost no difference in reading lecture notes between the two types of students. In fact, the higher performing students in Course#2 seem to become more active in reading lecture notes during the study period. Possibly, this is due to the fact that the level of knowledge in Course#2 tended to be less challenging than Course#1 and had less advanced concepts and examples. We speculate students' learning behaviours may change subject to the difficulty level of the study. We note

that Course#1 was designed for Software Engineering students and has a higher level of requirements for acquired knowledge and skills. Perhaps, the lower performing students in Course#1 might be struggling with acquiring new advanced concepts, which would keep them engaged longer with lecture notes instead of focusing on programming tasks.

With respect to the effect of the COVID19 pandemic, the learning behaviours of students were generally similar to the behaviours before the lockdown. The engagement of students in using learning material items tends to decrease over the semester, which is the same in all other cases in both courses. Furthermore, being highly active in practice-related items could be a signal of higher performing students. However, there is also a slight deviation in students' learning behaviours between the pre-COVID19 and during-COVID19 period due to the nature of studying from home during the lockdown. While lower performing students tend to read lecture notes more than higher performance students before the pandemic, this level of usage of lecture notes appears to be the same between the two cohorts during the pandemic. This observation may reflect that students might have limited communication opportunities and tend to merely rely on the given online learning materials.

In terms of the learning outcome prediction, using log data collected from online learning systems to predict students' success has been highly developed in the literature. There have been many scientific reports on building an early predicting system in various application contexts, from flagging "at-risk" students [24], to recommending next courses [169] and learning strategies [92, 17]. In our research, we provide a pre-processing data method that has been proven to be effective in improving the performance of widely used machine learning models in our context, i.e. programming education. This method can also be extended to different application contexts as shown above as long as the data satisfies one of the assumptions of Random Matrix Theory, e.g. the Q-factor is ensured. In addition, we have shown evidence that data features from learning behaviours could be good predictors to detect "at-risk" students in the early stage of the study period.

We recommend instructors to keep implementing community detection and pre-

diction as students' results come in. Other performance indicators can also be used in addition to lab exam grades, such as weekly exercise results. In practice, community detection can be implemented at any point during the study. Once communities are detected, the instructors can promptly implement interventions. For example, the higher performing groups can be given harder exercises to keep them focused and avoid getting bored with their studies. On the other hand, the lower performing groups should be given more basic tasks along with instructions or small-group tutor sessions. Furthermore, the instructors can provide lower performing communities with additional supporting materials or easier tasks with solutions. This would fill the knowledge gap and build up the confidence and motivation for the students as well as re-engage them in their studies.

6.3 Limitations

Although the proposed method appears to be successful in reflecting the relationship between students' learning behaviours and learning performance, there are limitations due to the assumptions of Random Matrix Theory - which might restrict the method from being applicable to all kinds of learning behavioural data. The distribution of eigenvalues is given by Equation 6 when the sample size (matrix rows) $m \to \infty$ and number of features (matrix column) $n \to \infty$, provided that the ratio of rows and columns is greater than or equal to 1. Hence, in the context of this thesis, the number of transitions extracted from event log data needs to be greater than the number of students.

There is also a concern in terms of using MST to reduce the size of the graph. When a distance matrix contains duplicate values, the associated graph will have duplicated edges. Consequently, it is possible for more than MST to be generated from the graph and thus the results of the analysis may not be stable. In such cases, other graph size reduction techniques can be considered to obtain a single reduced graph, ensuring the stability of results in further analysis. For example, the network sparsification technique can sparsifies the network while preserving network

structures and community properties [75]. The comparison between such techniques is out of the scope of this thesis and will be the target for future works in line with this research.

6.4 Conclusions

This thesis proposes a novel approach to analyse students' learning behaviour data in the context of programming education, using collected from an online learning system. The research is one of the first attempts to apply RMT and Community Detection in the educational domain. The analysis is based on a range of steps and techniques: (1) we extract a transition-student data matrix from the event log data; (2) we clean the effect of noise and trend in the correlation matrix of the transition-student data matrix, which is based on the Random Matrix Theory. This cleaning process can help to reveal the underlying meaning of the data; (3) the cleaned correlation matrix is used to construct a distance matrix and the Minimum Spanning Tree. The MST can represent the relationships of students' learning behaviours in using course material items in the form of an MST graph. Students having similar behaviours are found to be closer to each other in the constructed MST graph; (4) the Community Detection algorithms, i.e. Girvan Newman and Louvain, have been applied to detect the smaller student communities from the MST. Further educational contexts have also been considered, including the influence of COVID19 pandemic; (5) the student-event data matrix is also cleaned and used as input variables to predict the learning outcome of students in the lab exams, using a range of machine learning classification techniques. The findings from the above methods have been used to analyse the learning behaviours of students with different learning abilities in programming. The proposed approach in cleaning learning behavioural data also shows its effectiveness in Community Detection and building early prediction models. Insights from students' learning behaviours and recommendations are also discussed in this thesis.

In future work, we will also focus on changing the community structure (cur-

rently represented as an MST) of the students during the course. For example, a student may change their group in a different week, which may reveal that his or her learning behaviour also changes accordingly. This analysis could help to understand thoroughly how students study and to provide to better support for educators to improve the curriculum. However, this requires more advanced research approaches to be developed to process more complex data. The time duration on course material items will also be considered on top of the number of events in future works. Additionally, while it has been found that the community analysis results, using either the Girvan-Newman or Louvain method, do not vary significantly, the relationship between community detection techniques and analysis results is also worth further investigation, we believe. Regarding the processing of the learning behavioural dataset to remove the effect of noise and trend, more strategies to optimise the values of α and β will be investigated, which can help to improve the training of the predictive models. We will target these in future works for further deliver the insights of learning behaviours among student communities.

Bibliography

[1] Wil van der Aalst. "Process discovery from event data: Relating models and logs through abstractions". en. In: *WIREs Data Mining and Knowledge Discovery* 8.3 (2018), e1244. ISSN: 1942-4795.

[2] Wil van der Aalst. *Process Mining: Data Science in Action*. en. 2nd ed. 00461. Berlin Heidelberg: Springer-Verlag, 2016. ISBN: 978-3-662-49850-7.

[3] Wil van der Aalst and Ton Weijters. "Process mining: a research agenda". In: *Computers in Industry*. Process / Workflow Mining 53.3 (Apr. 2004). 00724, pp. 231–244. ISSN: 0166-3615.

[4] Wil van der Aalst et al. "Process Mining Manifesto". en. In: *Business Process Management Workshops*. Lecture Notes in Business Information Processing. 00628. Springer, Berlin, Heidelberg, Aug. 2011, pp. 169–194. ISBN: 978-3-642-28107-5. (Visited on 01/25/2018).

[5] Everaldo Aguiar et al. "Engagement vs performance: Using electronic portfolios to predict first semester engineering student retention". In: *ACM International Conference Proceeding Series* (Mar. 2014), pp. 103–112. DOI: 10.1145/2567574.2567583.

[6] A Patricia Aguilera-Hermida. "College students' use and acceptance of emergency online learning due to COVID-19". In: *International Journal of Educational Research Open* 1 (2020), p. 100011.

[7] Sunita B Aher and LMR Lobo. "Data preparation strategy in e-learning system using association rule algorithm". In: *International Journal of Computer Applications* 41.3 (2012).

[8] Hanan Aldowah, Hosam Al-Samarraie, and Wan Mohamad Fauzy. "Educational data mining and learning analytics for 21st century higher education: A review and synthesis". In: *Telematics and Informatics* 37 (2019), pp. 13–49.

[9] Norah Almusharraf and Shabir Khahro. "Students satisfaction with online learning experiences during the COVID-19 pandemic". In: *International Journal of Emerging Technologies in Learning (IJET)* 15.21 (2020), pp. 246–267.

[10] Abdullah Alsheddy and Mohamed Habib. "On the application of data mining algorithms for predicting student performance: A case study". In: *Int. J. Comput. Sci. Netw. Secur* 17.10 (2017), pp. 189–197.

[11] Aleksander Aristovnik et al. "Impacts of the COVID-19 pandemic on life of higher education students: A global perspective". In: *Sustainability* 12.20 (2020), p. 8438.

[12] Shadnaz Asgari et al. "An observational study of engineering online education during the COVID-19 pandemic". In: *PLoS One* 16.4 (2021), e0250041.

[13] Elizabeth Ayers, Rebecca Nugent, and Nema Dean. "A Comparison of Student Skill Knowledge Estimates". In: *Educational Data Mining* (2009), pp. 1–10.

[14] Ryan Baker et al. "Data mining for education". In: *International encyclopedia of education* 7.3 (2010), pp. 112–118.

[15] Ryan Baker and Adriana de Carvalho. "Labeling student behavior faster and more precisely with text replays". In: *Educational Data Mining 2008 - 1st International Conference on Educational Data Mining, Proceedings*. Jan. 2008, pp. 38–47.

[16] Behdad Bakhshinategh et al. “Educational data mining applications and tasks: A survey of the last 10 years”. en. In: *Education and Information Technologies* 23.1 (Jan. 2018), pp. 537–553. ISSN: 1573-7608.

[17] Abhinava Barthakur et al. “Assessing program-level learning strategies in MOOCs”. In: *Computers in Human Behavior* 117 (2021), p. 106674.

[18] Jaroslav Bayer et al. “Predicting Drop-Out from Social Behaviour of Students.” In: *5th International Conference on Educational Data Mining*. ERIC, 2012, pp. 103–109.

[19] Joseph E Beck. “Using learning decomposition to analyze student fluency development”. In: *ITS2006 Educational Data Mining Workshop*. 2006, pp. 21–28.

[20] Punam Bedi and Chhavi Sharma. “Community detection in social networks”. In: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 6.3 (2016), pp. 115–135.

[21] Mordechai Ben-Ari. “Constructivism in computer science education”. In: *J. of Computers in Mathematics and Science Teaching* 20.1 (2001), pp. 45–73.

[22] Galit Ben-Zadok et al. “Examining online learning processes based on log files analysis: A case study”. In: *5th International Conference on Multimedia and ICT in Education (m-ICTE’09)*. 2009.

[23] Jens Bennedsen and Michael E Caspersen. “Failure rates in introductory programming: 12 years later”. In: *ACM inroads* 10.2 (2019), pp. 30–36.

[24] Johannes Berens et al. “Early Detection of Students at Risk–Predicting Student Dropouts Using Administrative Student Data from German Universities and Machine Learning Methods.” In: *Journal of Educational Data Mining* 11.3 (2019), pp. 1–41.

[25] Sweta Bhattacharya et al. “A novel PCA-firefly based XGBoost classification model for intrusion detection in networks using GPU”. In: *Electronics* 9.2 (2020), p. 219.

[26] Paulo Blikstein. "Using learning analytics to assess students' behavior in open-ended programming tasks". In: *LAK '11: Proceedings of the 1st International Conference on Learning Analytics and Knowledge*. 2011, pp. 110–116.

[27] Vincent D Blondel et al. "Fast unfolding of communities in large networks". In: *Journal of statistical mechanics: theory and experiment* 2008.10 (2008), P10008.

[28] Alejandro Bogarín, Rebeca Cerezo, and Cristóbal Romero. "A survey on educational process mining". en. In: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 8.1 (Jan. 2018). ISSN: 1942-4795.

[29] RP Bose and Wil van der Aalst. "Trace clustering based on conserved patterns: Towards achieving better process models". In: *International Conference on Business Process Management*. Springer. 2009, pp. 170–181.

[30] Jean-Phillipe Bouchaud and Marc Potters. "Handbook on Random Matrix Theory". In: *Oxford University Press* (2011).

[31] Andrew W Braunstein, Mary N Lesser, and Donn R Pescatrice. "The Impact of a Program for the Disadvantaged on Student Retention." In: *College Student Journal* 42.1 (2008).

[32] Lars Buitinck et al. "API design for machine learning software: experiences from the scikit-learn project". In: *arXiv preprint arXiv:1309.0238* (2013).

[33] Joël Bun, Jean-Philippe Bouchaud, and Marc Potters. "Cleaning large correlation matrices:tools from random matrix theory". In: *Physics Reports* 666 (2017), pp. 1–109.

[34] Joël Bun and A Knowles. "An optimal rotational invariant estimator for general covariance matrices: The outliers". In: *Preprint* (2018).

[35] Kamal Bunkar et al. “Data mining: Prediction for performance improvement of graduate students using classification”. In: *2012 Ninth International Conference on Wireless and Optical Communications Networks (WOCN)*. IEEE. 2012, pp. 1–5.

[36] Awatef Cairns et al. “Process mining in the education domain”. In: *International Journal on Advances in Intelligent Systems* 8.1 (2015), pp. 219–232.

[37] Wagner L Cambruzzi, Sandro José Rigo, and Jorge LV Barbosa. “Dropout prediction and reduction in distance education courses with the learning analytics multitrail approach.” In: *J. Univers. Comput. Sci.* 21.1 (2015), pp. 23–47.

[38] Adam S Carter, Christopher D Hundhausen, and Olusola Adesope. “Blending measures of programming and social behavior into predictive models of student achievement in early computing courses”. In: *ACM Transactions on Computing Education (TOCE)* 17.3 (2017), pp. 1–20.

[39] Adam Scott Carter and Christopher David Hundhausen. “Using programming process data to detect differences in students’ patterns of programming”. In: *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education*. 2017, pp. 105–110.

[40] Kevin Casey and David Azcona. “Utilizing student activity patterns to predict performance”. In: *International Journal of Educational Technology in Higher Education* 14.1 (2017), pp. 1–15.

[41] Rebeca Cerezo et al. “Students’ LMS interaction patterns and their relationship with achievement: A case study in higher education”. In: *Computers & Education* 96 (2016), pp. 42–54.

[42] Mohamed Amine Chatti et al. “A reference model for learning analytics”. In: *International Journal of Technology Enhanced Learning* 4.5-6 (2012), pp. 318–331.

[43] Harshal Chaudhari and Martin Crane. "Cross-correlation dynamics and community structures of cryptocurrencies". In: *Journal of Computational Science* 44 (2020), p. 101130.

[44] Tianqi Chen and Carlos Guestrin. "XGBoost: A Scalable Tree Boosting System". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2016, pp. 785–794.

[45] Tianqi Chen et al. "Xgboost: extreme gradient boosting". In: *R package version 0.4-2* 1.4 (2015).

[46] Gregory S Chirikjian. "Multivariate statistical analysis and random matrix theory". In: *Stochastic Models, Information Theory, and Lie Groups, Volume 2*. Springer, 2012, pp. 229–270.

[47] KR1442 Chowdhary. "Natural language processing". In: *Fundamentals of artificial intelligence* (2020), pp. 603–649.

[48] Angel Cobo, Rocio Rocha, and Carlos Rodrıguez-Hoyos. "Evaluation of the interactivity of students in virtual learning environments using a multicriteria approach and data mining". In: *Behaviour & Information Technology* 33.10 (2014), pp. 1000–1012.

[49] Thomas Conlon, Heather J Ruskin, and Martin Crane. "Multiscaled cross-correlation dynamics in financial time-series". In: *Advances in Complex Systems* 12.04n05 (2009), pp. 439–454.

[50] Thomas Conlon, Heather J Ruskin, and Martin Crane. "Random matrix theory and fund of funds portfolio optimisation". In: *Physica A: Statistical Mechanics and its applications* 382.2 (2007), pp. 565–576.

[51] Corinna Cortes and Vladimir Vapnik. "Support-vector networks". In: *Machine learning* 20.3 (1995), pp. 273–297.

[52] Evandro B Costa et al. "Evaluating the effectiveness of educational data mining techniques for early prediction of students' academic failure in introductory programming courses". In: *Computers in Human Behavior* 73 (2017), pp. 247–256.

[53] Scott Crossley, Ran Liu, and Danielle McNamara. "Predicting math performance using natural language processing tools". In: *Proceedings of the Seventh International Learning Analytics & Knowledge Conference*. 2017, pp. 339–347.

[54] Juan Cruz-Benito et al. "Discovering usage behaviors and engagement in an Educational Virtual World". In: *Computers in Human Behavior* 47 (2015), pp. 18–25.

[55] Padraig Cunningham and Sarah Jane Delany. "k-Nearest neighbour classifiers-A Tutorial". In: *ACM Computing Surveys (CSUR)* 54.6 (2021), pp. 1–25.

[56] Justin Daly, Martin Crane, and Heather J Ruskin. "Random matrix theory filters in portfolio optimisation: A stability and risk assessment". In: *Physica A: Statistical Mechanics and its Applications* 387.16-17 (2008), pp. 4248–4260.

[57] Gerben W Dekker, Mykola Pechenizkiy, and Jan M Vleeshouwers. "Predicting Students Drop Out: A Case Study." In: *Computers, Environment and Urban Systems* (2009), pp. 41–50.

[58] Tenzin Doleck et al. "Examining diagnosis paths: A process mining approach". In: *2016 Second International Conference on Computational Intelligence & Communication Technology (CICT)*. IEEE. 2016, pp. 663–667.

[59] Laurie P Dringus and Timothy Ellis. "Using data mining as a strategy for assessing asynchronous discussion forums". In: *Computers & Education* 45.1 (2005), pp. 141–160.

[60] Xu Du et al. “Educational data mining: a systematic review of research and emerging trends”. In: *Information Discovery and Delivery* 48.4 (2020), pp. 225–236.

[61] Freeman J Dyson. “Distribution of eigenvalues for a class of real symmetric matrices”. In: *Revista Mexicana de Fisica* 20.4 (1971), pp. 231–237.

[62] Freeman J Dyson. “Statistical theory of the energy levels of complex systems. I”. In: *Journal of Mathematical Physics* 3.1 (1962), pp. 140–156.

[63] James E Eckles and Eric G Stradley. “A social network analysis of student retention using archival data”. In: *Social Psychology of Education* 15.2 (2012), pp. 165–180.

[64] Varick L Erickson. “Data-driven models to predict student performance and improve advising in computer science”. In: *Proceedings of the International Conference on Frontiers in Education: Computer Science and Computer Engineering (FECS)*. 2019, pp. 3–9.

[65] Rebecca Eynon and Lars-Erik Malmberg. “Lifelong learning and the Internet: Who benefits most from learning online?” In: *British Journal of Educational Technology* 52.2 (2021), pp. 569–583.

[66] Daniel Amo Filvà et al. “Clickstream for learning analytics to assess students’ behavior with Scratch”. In: *Future Generation Computer Systems* 93 (2019), pp. 673–686.

[67] Jerome H Friedman. “Greedy function approximation: a gradient boosting machine”. In: *Annals of statistics* (2001), pp. 1189–1232.

[68] G Gajewski, J Chołoniewski, and JA Hołyst. “Key courses of academic curriculum uncovered by data mining of students’ grades”. In: *Acta Physica Polonica A* 129.5 (2016), pp. 1071–1076.

[69] Anna Karen Gárate-Escamila, Amir Hajjam El Hassani, and Emmanuel Andrès. “Classification models for heart disease prediction using feature selection and PCA”. In: *Informatics in Medicine Unlocked* 19 (2020), p. 100330.

[70] Gartner. *E-learning*. en. URL: https://www.gartner.com/en/information-technology/glossary/e-learning (visited on 02/20/2020).

[71] Dragan Gašević et al. "Learning analytics should not promote one size fits all: The effects of instructional conditions in predicting academic success". In: *The Internet and Higher Education* 28 (2016), pp. 68–84.

[72] Mohamed A Ghazal, Osman Ibrahim, and Mostafa A Salama. "Educational process mining: a systematic literature review". In: *2017 European Conference on Electrical Engineering and Computer Science (EECS)*. IEEE. 2017, pp. 198–203.

[73] Graham Gibbs and Claire Simpson. "Conditions Under Which Assessment Supports Students' Learning". en. In: *Learning and Teaching in Higher Education* 1 (2005), pp. 3–31. ISSN: 1742-240X.

[74] Brittany Gilbert. "Online learning revealing the benefits and challenges". In: *Education Masters* Paper 303 (2015).

[75] Aristides Gionis et al. "Community-aware network sparsification". In: *Proceedings of the 2017 SIAM International Conference on Data Mining*. SIAM. 2017, pp. 426–434.

[76] Michelle Girvan and Mark EJ Newman. "Community structure in social and biological networks". In: *Proceedings of the national academy of sciences* 99.12 (2002), pp. 7821–7826.

[77] Teresa Gonzalez et al. "Influence of COVID-19 confinement on students' performance in higher education". In: *PloS one* 15.10 (2020), e0239490.

[78] John C Gower, Sugnet Gardner Lubbe, and Niel J Le Roux. *Understanding biplots*. John Wiley & Sons, 2011. ISBN: 978-0-470-01255-0.

[79] Christian W. Günther and Wil van der Aalst. "Fuzzy Mining – Adaptive Process Simplification Based on Multi-perspective Metrics". en. In: *Business Process Management*. Ed. by David Hutchison et al. Vol. 4714. 00000. Berlin,

Heidelberg: Springer Berlin Heidelberg, 2007, pp. 328–343. ISBN: 978-3-540-75182-3.

[80] LR Haff. "Empirical Bayes estimation of the multivariate normal covariance matrix". In: *The Annals of Statistics* (1980), pp. 586–597.

[81] W Hämäläinen, TH Laine, and E Sutinen. "Data mining in personalizing distance education courses". In: *Data mining in e-learning* (2006), pp. 157–171.

[82] Jiawei Han, Jian Pei, and Micheline Kamber. *Data mining: concepts and techniques*. Elsevier, 2011. ISBN: 978-0-12-381479-1.

[83] Najmul Hasan and Yukun Bao. "Impact of "e-Learning crack-up" perception on psychological distress among college students during COVID-19 pandemic: A mediating role of "fear of academic year loss"". In: *Children and Youth Services Review* 118 (2020), p. 105355.

[84] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. "Model assessment and selection". In: *The elements of statistical learning*. Springer, 2009, pp. 219–259.

[85] Wu He. "Examining students' online interaction in a live video streaming environment using data mining and text mining". In: *Computers in Human Behavior* 29.1 (2013), pp. 90–102.

[86] Cecily Heiner, Joseph Beck, and Jack Mostow. "Lessons on using ITS data to answer educational research questions". In: *Proceedings of the workshop Analyzing Student-Tutor Interaction Logs to Improve Educational Outcomes at ITS*. 2004, pp. 1–9.

[87] Mohammad Hossin and Md Nasir Sulaiman. "A review on evaluation metrics for data classification evaluations". In: *International journal of data mining & knowledge management process* 5.2 (2015), p. 1.

[88] Jin Huang and Charles X Ling. "Using AUC and accuracy in evaluating learning algorithms". In: *IEEE Transactions on knowledge and Data Engineering* 17.3 (2005), pp. 299–310.

[89] Jui-Long Hung and Ke Zhang. "Revealing online learning behaviors and activity patterns and making predictions with data mining techniques in online teaching". In: *MERLOT Journal of Online Learning and Teaching* 4.4 (2008), pp. 426–437.

[90] Jui-Long Hung et al. "Identifying at-risk students for early interventions: A time-series clustering approach". In: *IEEE Transactions on Emerging Topics in Computing* 5.1 (2015), pp. 45–55.

[91] Wu-Yuin Hwang et al. "A pilot study of cooperative programming learning behavior and its relationship with students' learning performance". In: *Computers & education* 58.4 (2012), pp. 1267–1281.

[92] Jelena Jovanović et al. "Supporting actionable intelligence: reframing the analysis of observed study strategies". In: *Proceedings of the Tenth International Conference on Learning Analytics & Knowledge*. 2020, pp. 161–170.

[93] Mehmed Kantardzic. *Data mining: concepts, models, methods, and algorithms*. John Wiley & Sons, 2011. ISBN: 978-0-470-89045-5.

[94] Stephen Kelley et al. "Defining and discovering communities in social networks". In: *Handbook of Optimization in Complex Networks*. Springer, 2012, pp. 139–168.

[95] Dong-Hee Kim and Hawoong Jeong. "Systematic analysis of group identification in stock markets". In: *Physical Review E* 72.4 (2005), p. 046133.

[96] David A Kolb. *Experiential learning: Experience as the source of learning and development*. FT press, 2014. ISBN: 978-0-13-389240-6.

[97] Joseph B Kruskal. "On the shortest spanning subtree of a graph and the traveling salesman problem". In: *Proceedings of the American Mathematical society* 7.1 (1956), pp. 48–50.

[98] Laurent Laloux et al. “Random matrix theory and financial correlations”. In: *International Journal of Theoretical and Applied Finance* 3.03 (2000), pp. 391–397.

[99] Andrea Lancichinetti and Santo Fortunato. “Community detection algorithms: a comparative analysis”. In: *Physical review E* 80.5 (2009), p. 056117.

[100] Marcus A Lashley et al. “How the ecology and evolution of the COVID-19 pandemic changed learning”. In: *Ecology and Evolution* 10.22 (2020).

[101] Roderick JA Little and Donald B Rubin. *Statistical analysis with missing data*. Vol. 793. John Wiley & Sons, 2019.

[102] Bing Liu. *Web data mining: exploring hyperlinks, contents, and usage data*. Vol. 1. Springer, 2011.

[103] Jing Luan. “Data Mining and Knowledge Management in Higher Education-Potential Applications.” In: *46th Annual Forum for the Association, for Institutional Research*. ERIC, 2002, p. 19.

[104] Ioanna Lykourentzou et al. “Early and dynamic student achievement prediction in e-learning courses using neural networks”. In: *Journal of the American Society for Information Science and Technology* 60.2 (2009), pp. 372–380.

[105] Tai Tan Mai, Martin Crane, and Marija Bezbradica. “Students’ Behaviours in using Learning Resources in Higher Education: How do behaviours reflect success in Programming Education?” In: *Proceedings of the 7th International Conference on Higher Education Advances (HEAd’21)*. 2021, pp. 47–55.

[106] Donia Malekian, James Bailey, and Gregor Kennedy. “Prediction of Students’ Assessment Readiness in Online Learning Environments: The Sequence Matters”. en. In: *LAK ’20: Proceedings of the Tenth International Conference on Learning Analytics & Knowledge*. 2020, pp. 382–391.

[107] Tatiana Markova, Irina Glazkova, and Elena Zaborova. “Quality issues of online distance learning”. In: *Procedia-Social and Behavioral Sciences* 237 (2017), pp. 685–691.

[108] Judi McCuaig and Julia Baldwin. "Identifying Successful Learners from Interaction Behaviour." In: *Proceeding of 5th International Conference on Educational Data Mining*. ERIC, 2012, p. 4.

[109] Alves de Medeiros, Ton Weijters, and Wil van der Aalst. "Using genetic algorithms to mine process models: representation, operators and results". In: Eindhoven: Technische Universiteit Eindhoven, 2004. ISBN: 90-386-2287-2.

[110] Rodrigo Pessoa Medeiros, Geber Lisboa Ramalho, and Taciana Pontual Falcão. "A systematic literature review on teaching and learning introductory programming in higher education". In: *IEEE Transactions on Education* 62.2 (2018), pp. 77–90.

[111] Madan Lal Mehta and Freeman J Dyson. "Statistical theory of the energy levels of complex systems. V". In: *Journal of Mathematical Physics* 4.5 (1963), pp. 713–719.

[112] Agathe Merceron and Kalina Yacef. "Educational Data Mining: a Case Study." In: *Artificial Intelligence in Education*. IOS Press, 2005, pp. 467–474.

[113] Francesco Mezzadri, Nina C Snaith, NJ Hitchin, et al. *Recent perspectives in random matrix theory and number theory*. Vol. 322. Cambridge University Press, 2005. ISBN: 978-0-521-62058-1.

[114] Sein Minn et al. "Deep Knowledge Tracing and Dynamic Student Classification for Knowledge Tracing". In: *2018 IEEE International Conference on Data Mining (ICDM)*. ISSN: 1550-4786. Nov. 2018, pp. 1182–1187.

[115] Pedro Manuel Moreno-Marcos et al. "Prediction in MOOCs: A review and future research directions". In: *IEEE Transactions on Learning Technologies* 12.3 (2018), pp. 384–401.

[116] Kew Si Na and Zaidatun Tasir. "Identifying at-risk students in online learning by analysing learning behaviour: A systematic review". In: *2017 IEEE Conference on Big Data and Analytics (ICBDA)*. IEEE. 2017, pp. 118–123.

[117] Nadim Nachar et al. “The Mann-Whitney U: A test for assessing whether two independent samples come from the same distribution”. In: *Tutorials in quantitative Methods for Psychology* 4.1 (2008), pp. 13–20.

[118] John Neter et al. “Applied linear statistical models”. In: *Journal of the American Statistical Association* 103 (2008), pp. 880–880.

[119] Mark EJ Newman and Michelle Girvan. “Finding and evaluating community structure in networks”. In: *Physical review E* 69.2 (2004), p. 026113.

[120] Karthik Nilakant and Antonija Mitrovic. “Applications of Data Mining in Constraint-based Intelligent Tutoring Systems”. In: *Proceedings of the 2005 conference on Artificial Intelligence in Education: Supporting Learning through Intelligent and Socially Informed Technology*. 2005, pp. 896–898.

[121] Clara Nkhoma et al. “Learning analytics techniques and visualisation with textual data for determining causes of academic failure”. In: *Behaviour & Information Technology* 39.7 (2020), pp. 808–823.

[122] OMG. “Business Process Model and Notation (BPMN): Version 2.0 specification”. In: *Object Management Group* Tech. Rep. formal/2011-01-03 (Jan. 2011). 00000.

[123] Maria Pampaka, Graeme Hutcheson, and Julian Williams. “Handling missing data: analysis of a challenging data set using multiple imputation”. In: *International Journal of Research & Method in Education* 39.1 (2016), pp. 19–37.

[124] Zacharoula K Papamitsiou and Anastasios A Economides. “Learning analytics and educational data mining in practice: A systematic literature review of empirical evidence.” In: *J. Educ. Technol. Soc.* 17.4 (2014), pp. 49–64.

[125] Suhem Parack, Zain Zahid, and Fatima Merchant. “Application of data mining in educational databases for predicting academic trends and patterns”. In: *2012 IEEE international conference on technology enhanced education (ICTEE)*. IEEE. 2012, pp. 1–4.

[126] Zachary A Pardos et al. “Adapting Bayesian Knowledge Tracing to a Massive Open Online Course in edX”. In: *Proceedings of the 6th International Conference on Educational Data Mining*. 2013.

[127] Terry Peckham and Gord McCalla. “Mining Student Behavior Patterns in Reading Comprehension Tasks.” In: *Proceeding of 5th International Conference on Educational Data Mining*. ERIC, 2012, p. 8.

[128] F. Pedregosa et al. “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[129] Aleksandar Pejić and Piroska Stanić Molcer. “Exploring data mining possibilities on computer based problem solving data”. In: *2016 IEEE 14th International Symposium on Intelligent Systems and Informatics (SISY)*. IEEE. 2016, pp. 171–176.

[130] Alejandro Peña-Ayala. “Educational data mining: A survey and a data mining-based analysis of recent works”. In: *Expert systems with applications* 41.4 (2014), pp. 1432–1462.

[131] Dilhan Perera et al. “Clustering and sequential pattern mining of online collaborative learning data”. In: *IEEE Transactions on Knowledge and Data Engineering* 21.6 (2008), pp. 759–772.

[132] Chris Piech et al. “Deep Knowledge Tracing”. In: *Advances in Neural Information Processing Systems 28*. Ed. by C. Cortes et al. Curran Associates, Inc., 2015, pp. 505–513.

[133] Vasiliki Plerou et al. “Random matrix approach to cross correlations in financial data”. In: *Physical Review E* 65.6 (2002), p. 066126.

[134] Anjana Pradeep, Smija Das, and Jubilant J Kizhekkethottam. “Students dropout factor prediction using EDM techniques”. In: *2015 International Conference on Soft-Computing and Networks Security (ICSNS)*. IEEE. 2015, pp. 1–7.

[135] Marcos M López de Prado. *Machine learning for asset managers*. Cambridge University Press, 2020. ISBN: 978-1-10-888365-8.

[136] Qasem A Al-Radaideh, Emad M Al-Shawakfa, and Mustafa I Al-Najjar. "Mining student data using decision trees". In: *International Arab Conference on Information Technology (ACIT'2006), Yarmouk University, Jordan*. 2006.

[137] Filippo Radicchi et al. "Defining and identifying communities in networks". In: *Proceedings of the national academy of sciences* 101.9 (2004), pp. 2658–2663.

[138] Masura Rahmat et al. "Major problems in basic programming that influence student performance". In: *Procedia-Social and Behavioral Sciences* 59 (2012), pp. 287–296.

[139] Robert Redpath and Judy Sheard. "Domain knowledge to support understanding and treatment of outliers". In: *International Conference on Information and Automation*. Citeseer. 2005, pp. 398–403.

[140] Peter Reimann, Jimmy Frerejean, and Kate Thompson. "Using process mining to identify models of group decision making in chat data". In: *Computer Supported Collaborative Learning Practices: CSCL2009 Conference Proceedings*. International Society of the Learning Sciences (ISLS), 2009, pp. 98–107.

[141] Qian Ren et al. "Network Modelling and Visualisation Analysis of the Undergraduate Dental Curriculum System in China". In: *Journal of Computer and Communications* 9.6 (2021), pp. 38–51.

[142] S Retalis et al. "Towards networked learning analytics–A concept and a tool". In: *Proceedings of the fifth international conference on networked learning*. 2006, pp. 1–8.

[143] Marcos Wander Rodrigues, Seiji Isotani, and Luiz Enrique Zarate. "Educational Data Mining: A review of evaluation process in the e-learning". In: *Telematics and Informatics* 35.6 (2018), pp. 1701–1717.

[144] Mojibur Rohman et al. "Online learning in higher education during covid-19 pandemic: students' perceptions". In: *Journal of Talent Development and Excellence* 12.2s (2020), pp. 3644–3651.

[145] Cristobal Romero and Sebastian Ventura. "Data mining in education". In: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 3.1 (2013), pp. 12–27.

[146] Cristobal Romero and Sebastian Ventura. "Educational data mining: A survey from 1995 to 2005". In: *Expert systems with applications* 33.1 (2007), pp. 135–146.

[147] Cristóbal Romero, José Raúl Romero, and Sebastián Ventura. "A survey on pre-processing educational data". In: *Educational data mining*. Springer, 2014, pp. 29–64.

[148] Cristóbal Romero and Sebastián Ventura. "Educational data mining: a review of the state of the art". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 40.6 (2010), pp. 601–618.

[149] Cristóbal Romero, Sebastián Ventura, and Enrique Garcıa. "Data mining in course management systems: Moodle case study and tutorial". In: *Computers & Education* 51.1 (2008), pp. 368–384.

[150] Cristóbal Romero et al. "Association rule mining using genetic programming to provide feedback to instructors from multiple-choice quiz data". In: *Expert Systems* 30.2 (2013), pp. 162–172.

[151] Cristóbal Romero et al. "Educational process mining: A tutorial and case study using moodle data sets". In: *Data mining and learning analytics: Applications in educational research* 1 (2016).

[152] Cristóbal Romero et al. “Mining rare association rules from e-learning data”. In: *Proceedings of International Conference on Educational Data Mining*. ERIC. 2010.

[153] Andrew Rosenberg and Julia Hirschberg. “V-measure: A conditional entropy-based external cluster evaluation measure”. In: *Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL)*. 2007, pp. 410–420.

[154] Daniel J Salas et al. “Supporting the acquisition of scientific skills by the use of learning analytics”. In: *International Conference on Web-Based Learning*. Springer. 2016, pp. 281–293.

[155] Bilal Saoud and Abdelouahab Moussaoui. “Community detection in networks based on minimum spanning tree and modularity”. In: *Physica A: Statistical Mechanics and its Applications* 460 (2016), pp. 230–234.

[156] Gayane Sedrakyan, Jochen De Weerdt, and Monique Snoeck. “Process-mining enabled feedback: “Tell me what I did wrong” vs. “tell me how to do it right””. en. In: *Computers in Human Behavior* 57 (Apr. 2016), pp. 352–376. ISSN: 0747-5632.

[157] Helena Seli, Myron H. Dembo, and Myron H. Dembo. *Motivation and Learning Strategies for College Success : A Focus on Self-Regulated Learning*. en. Routledge, Dec. 2012. ISBN: 978-0-203-81383-6.

[158] Raghad Al-Shabandar et al. “Machine learning approaches to predict learning outcomes in Massive open online courses”. In: *2017 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2017, pp. 713–720.

[159] Dalia Abdulkareem Shafiq et al. “Student Retention using Educational Data Mining and Predictive Analytics: A Systematic Literature Review”. In: *IEEE Access* (2022).

[160] Saba Sharifi et al. “Random matrix theory for portfolio optimization: a stability approach”. In: *Physica A: Statistical Mechanics and its Applications* 335.3-4 (2004), pp. 629–643.

[161] Kshitij Sharma et al. “Predicting learners’ effortful behaviour in adaptive assessment using multimodal data”. In: *Proceedings of the Tenth International Conference on Learning Analytics & Knowledge*. 2020, pp. 480–489.

[162] Ruey-Shiang Shaw. “A study of the relationships among learning styles, participation types, and performance in programming language learning supported by online forums”. In: *Computers & Education* 58.1 (2012), pp. 111–120.

[163] George Siemens and Phil Long. “Penetrating the fog: Analytics in learning and education.” In: *EDUCAUSE review* 46.5 (2011), p. 30.

[164] Leonardo Silva et al. “Regulation of learning interventions in programming education: A systematic literature review and guideline proposition”. In: *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education*. 2021, pp. 647–653.

[165] Jean Simon. “Data preprocessing using a priori knowledge”. In: *The 6th International Conference on Educational Data Mining (EDM 2013)*. 2016.

[166] Katrina Sin and Loganathan Muthu. “Application of Big Data in Education Data Mining and Learning Analytics–A Literature Review.” In: *ICTACT journal on soft computing* 5.4 (2015).

[167] *Student Central — usc.custhelp.com*. en. URL: `https://usc.custhelp.com/app/answers/detail/a_id/1888/kw/difference%5C%20between%5C%20course%5C%20and%5C%20program/` (visited on 02/18/2020).

[168] Gengxin Sun and Sheng Bin. “Topic Interaction Model Based on Local Community Detection in MOOC Discussion Forums and its Teaching”. In: *Educational Sciences: Theory & Practice* 18.6 (2018).

[169] Mack Sweeney et al. “Next-Term Student Performance Prediction: A Recommender Systems Approach”. In: *Journal of Educational Data Mining* 8.1 (2016), pp. 22–51.

[170] Vincent A Traag, Ludo Waltman, and Nees Jan Van Eck. “From Louvain to Leiden: guaranteeing well-connected communities”. In: *Scientific reports* 9.1 (2019), pp. 1–12.

[171] Nghi Truong, Peter Bancroft, and Paul Roe. “A web based environment for learning to program”. In: *Proceedings of the 26th Australasian computer science conference-Volume 16*. 2003, pp. 255–264.

[172] Antonia M Tulino, Sergio Verdú, and Sergio Verdu. *Random matrix theory and wireless communications*. Now Publishers Inc, 2004.

[173] Angela Van Barneveld, Kimberly E Arnold, and John P Campbell. “Analytics in higher education: Establishing a common language”. In: *EDUCAUSE learning initiative* 1.1 (2012), pp. l–ll.

[174] Wil Van Der Aalst. “Process mining”. In: *Communications of the ACM* 55.8 (2012), pp. 76–83.

[175] Kuo-Hua Wang et al. “Learning styles and formative assessment strategy: enhancing student achievement in Web-based learning”. In: *Journal of computer assisted learning* 22.3 (2006), pp. 207–217.

[176] Huan Wei et al. “Predicting student performance in interactive online question pools using mouse interaction features”. In: *Proceedings of the Tenth International Conference on Learning Analytics & Knowledge*. 2020, pp. 645–654.

[177] Ton Weijters and Wil van der Aalst. “Rediscovering workflow models from event-based data using little thumb”. en. In: *Integrated Computer-Aided Engineering* 10.2 (May 2003). 00529, pp. 151–162. ISSN: 18758835, 10692509.

[178] Ton Weijters and Wil van der Aalst. "Workflow mining: discovering workflow models from event-based data". English. In: *Proceedings of the ECAI Workshop on Knowledge Discovery and Spatial Data* (2002). 00058, pp. 78–84. (Visited on 12/17/2019).

[179] Douglas Brent West et al. *Introduction to graph theory*. Vol. 2. Prentice hall Upper Saddle River, 2001.

[180] Eugene P Wigner. "On the statistical distribution of the widths and spacings of nuclear resonance levels". In: *Mathematical Proceedings of the Cambridge Philosophical Society*. Vol. 47. 4. Cambridge University Press. 1951, pp. 790–798.

[181] John Wishart. "The generalised product moment distribution in samples from a normal multivariate population". In: *Biometrika* (1928), pp. 32–52.

[182] Gary KW Wong and Simon YK Li. "Academic performance prediction using chance discovery from online discussion forums". In: *2016 IEEE 40th Annual Computer Software and Applications Conference (COMPSAC)*. Vol. 1. IEEE. 2016, pp. 706–711.

[183] Sen Wu, Mengjiao Tuo, and Deying Xiong. "Community structure detection of shanghai stock market based on complex networks". In: *LISS 2014*. Springer, 2015, pp. 1661–1666.

[184] Wanli Xing et al. "Participation-based student final performance prediction model through interpretable Genetic Programming: Integrating learning analytics, educational data mining and theory". In: *Computers in Human Behavior* 47 (2015), pp. 168–181.

[185] Jaewon Yang, Julian McAuley, and Jure Leskovec. "Community detection in networks with node attributes". In: *2013 IEEE 13th international conference on data mining*. IEEE. 2013, pp. 1151–1156.

[186] Stephen JH Yang et al. "Predicting students' academic performance using multiple linear regression and principal component analysis". In: *Journal of Information Processing* 26 (2018), pp. 170–176.

[187] Xu Yang et al. "Learner behaviors in synchronous online prosthodontic education during the 2020 COVID-19 pandemic". In: *The Journal of prosthetic dentistry* 126.5 (2020), pp. 653–657.

[188] Ding Yanrui et al. "Identifying the Communities in the Metabolic Network Using'Component'Definition and Girvan-Newman Algorithm". In: *2015 14th International Symposium on Distributed Computing and Applications for Business Engineering and Science (DCABES)*. IEEE. 2015, pp. 42–45.

[189] Sahar Yassine, Seifedine Kadry, and Miguel-Angel Sicilia. "Application of community detection algorithms on learning networks. The case of Khan Academy repository". In: *Computer Applications in Engineering Education* 29.2 (2021), pp. 411–424.

[190] Michael Yudelson, Roya Hosseini, and Peter Brusilovsky. "Investigating Automated Student Modeling in a Java MOOC". en. In: *Proceeding of International Conference on Educational Data Mining*. 2014, pp. 261–264.

[191] Muhammed Yusuf. "The impact of self-efficacy, achievement motivation, and self-regulated learning strategies on students' academic achievement". In: *Procedia - Social and Behavioral Sciences*. 3rd World Conference on Educational Sciences - 2011 15 (2011), pp. 2623–2626.

[192] Xiao Zhang et al. "A review of community detection algorithms based on modularity optimization". In: *Journal of Physics: Conference Series*. Vol. 1069. 1. IOP Publishing. 2018, p. 012123.

Appendix A

Community Detection by Girvan Newman algorithm represented as Minimum Spanning Tree graphs

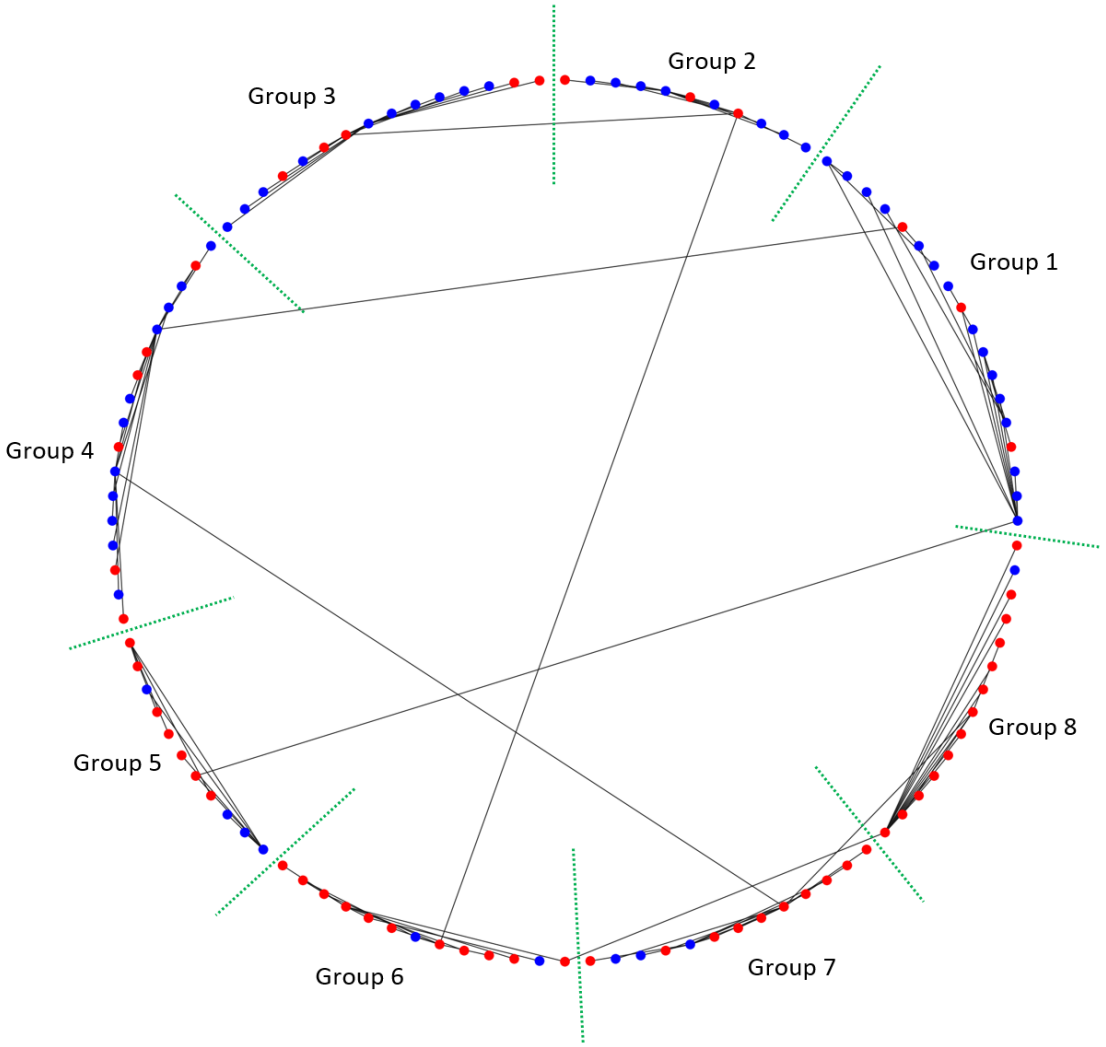


Figure A.1: Detected communities in Course#1-2018 represented as a Minumum Spanning Tree graph. Blue dots refer to higher performing students; Red dots refer to lower performing students.

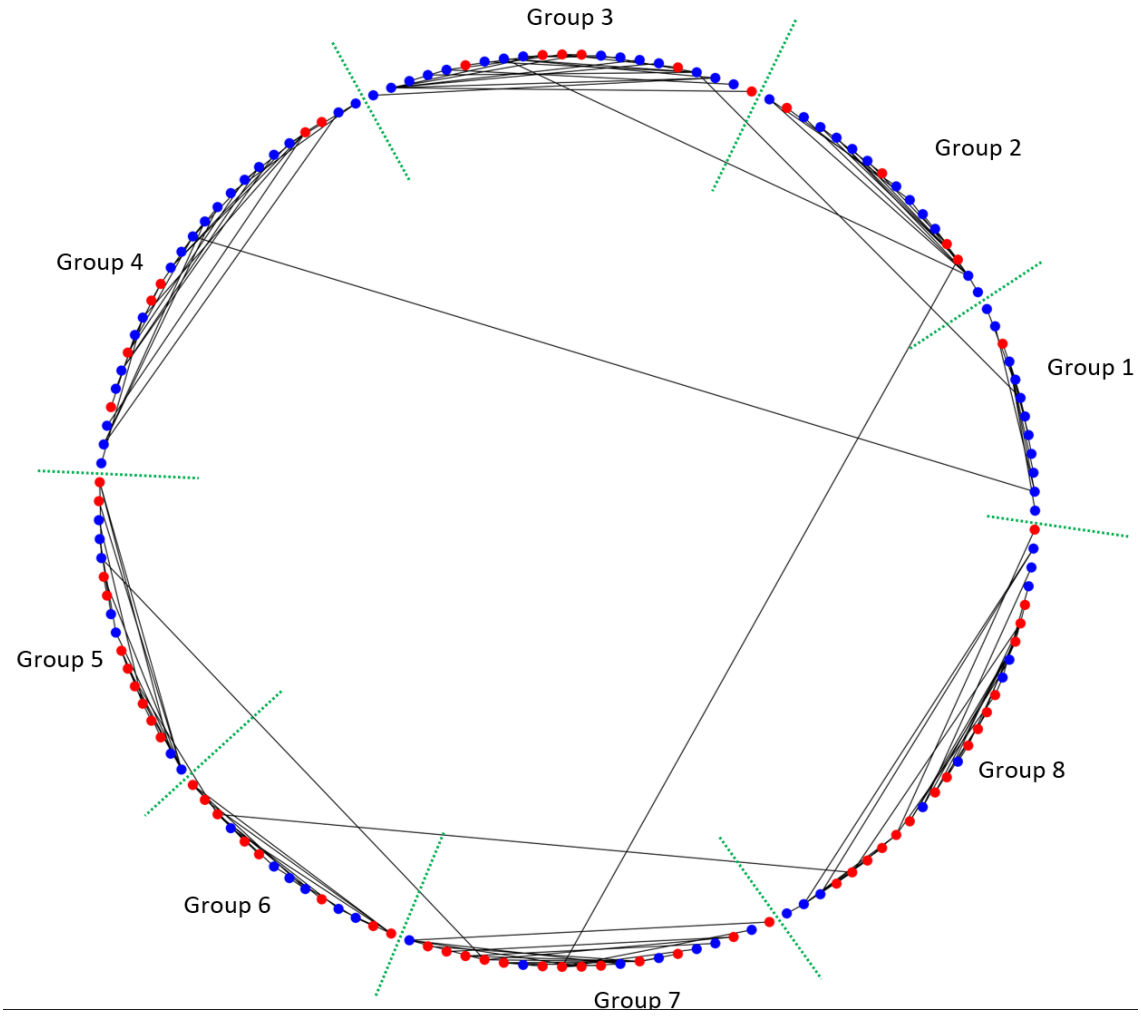


Figure A.2: Detected communities in Course#1-2019 represented as a Minumum Spanning Tree graph. Blue dots refer to higher performing students; Red dots refer to lower performing students.

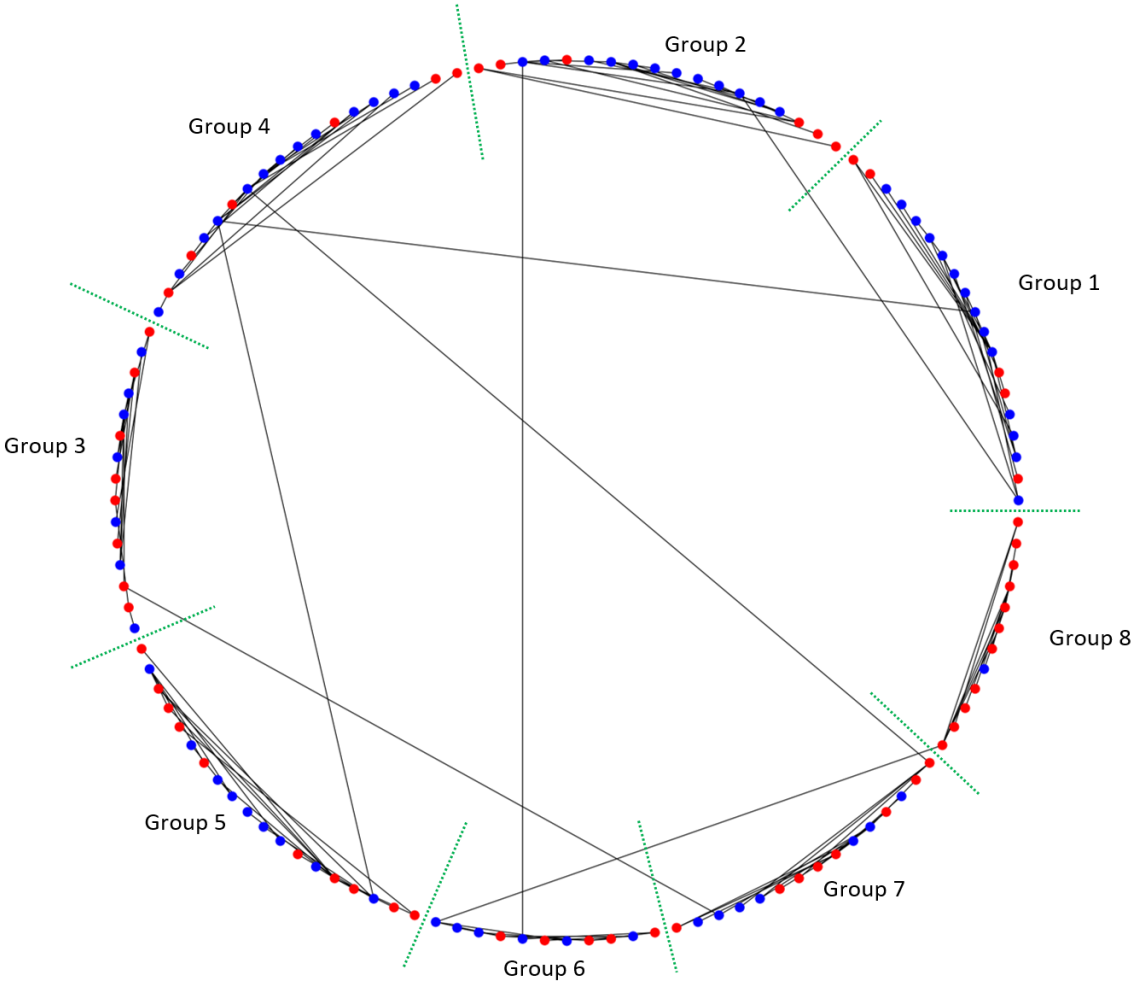


Figure A.3: Detected communities in Course#1-2020 represented as a Minumum Spanning Tree graph. Blue dots refer to higher performing students; Red dots refer to lower performing students.

Appendix B

Community Detection by Girvan Newman algorithm represented as dendrograms

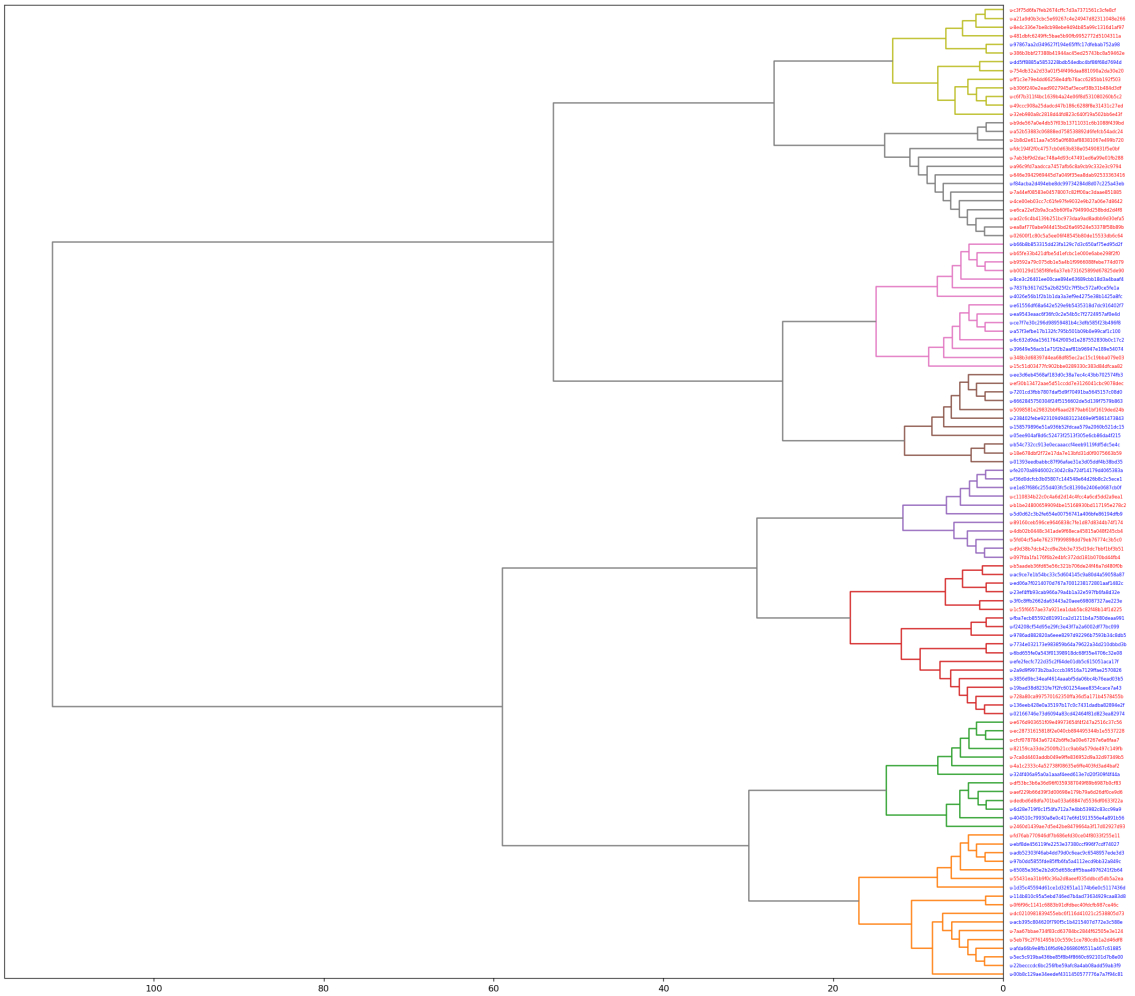


Figure B.1: Detected communities in Course#1-2018 represented as a dendrogram. Blue labels refer to higher performing students; Red labels refer to lower performing students.

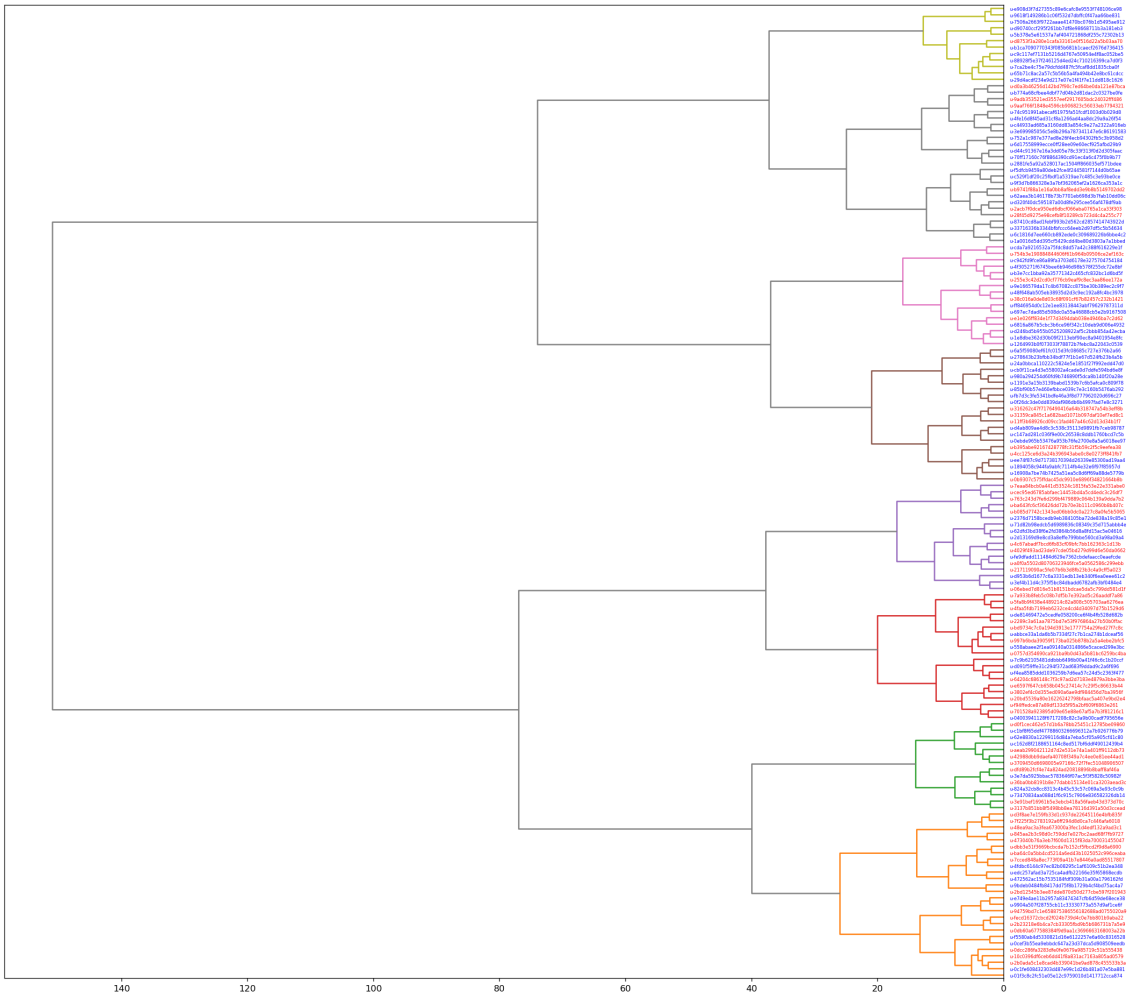


Figure B.2: Detected communities in Course#1-2019 represented as a dendrogram. Blue labels refer to higher performing students; Red labels refer to lower performing students.

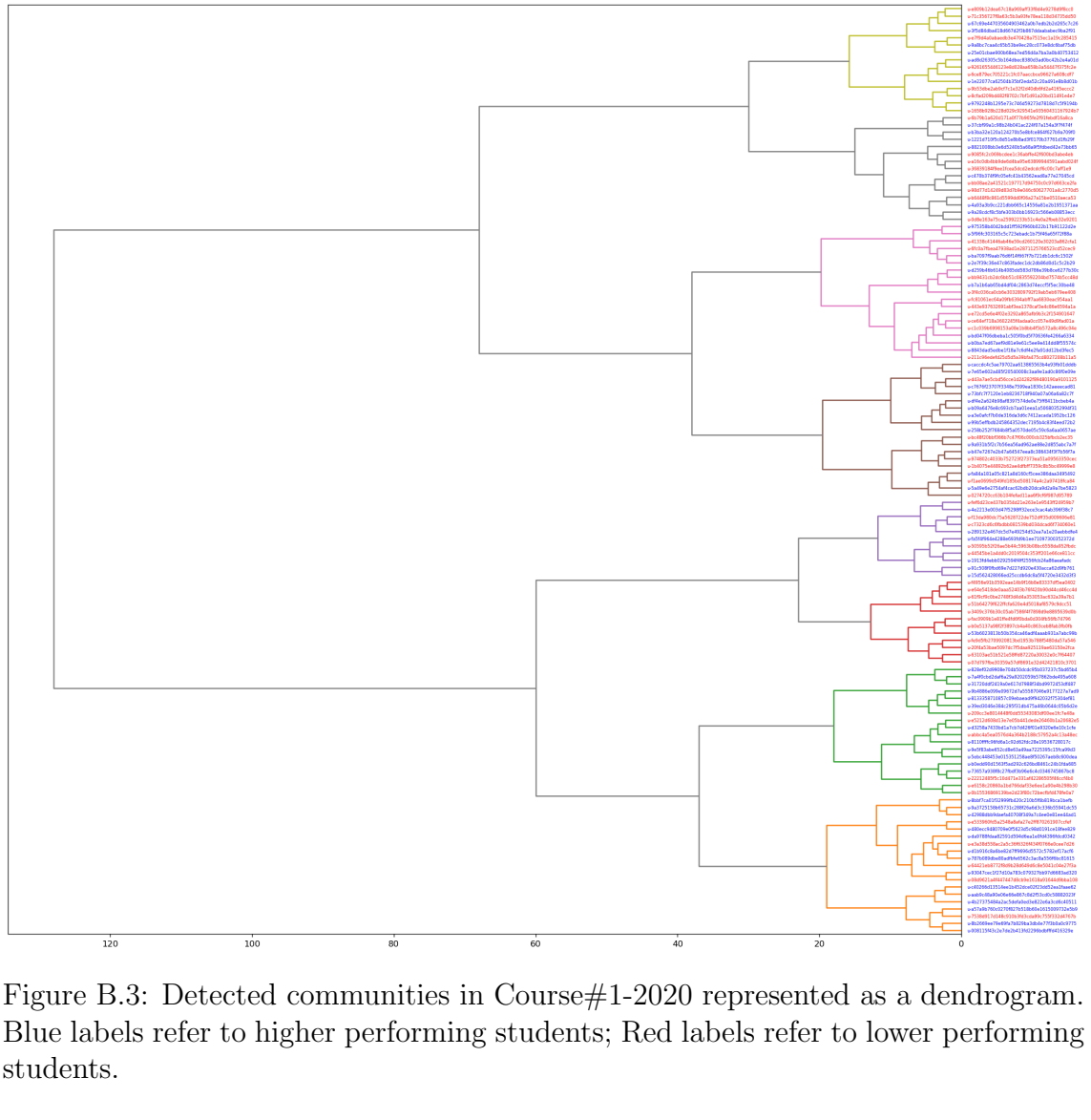


Figure B.3: Detected communities in Course#1-2020 represented as a dendrogram. Blue labels refer to higher performing students; Red labels refer to lower performing students.

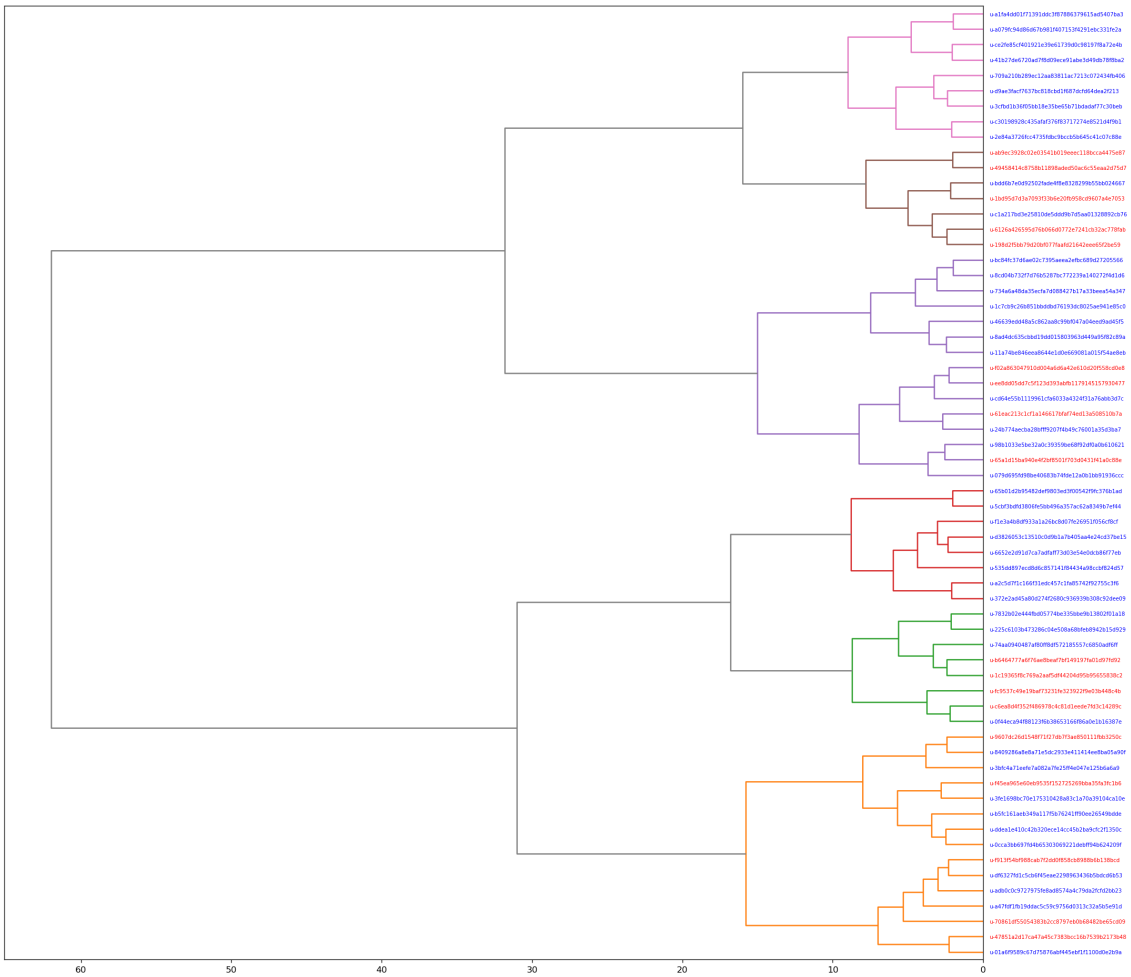


Figure B.4: Detected communities in Course#2-2018 represented as a dendrogram. Blue labels refer to higher performing students; Red labels refer to lower performing students.

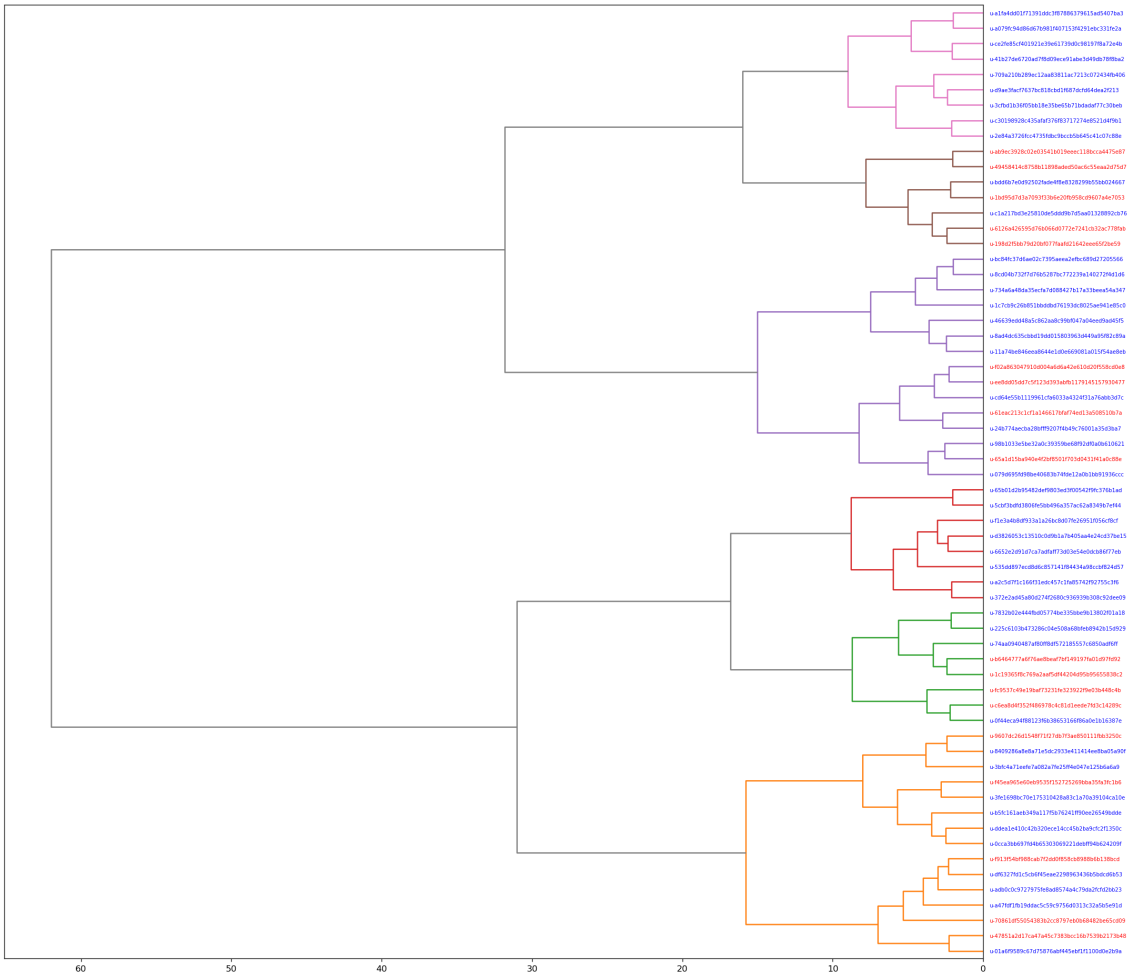


Figure B.5: Detected communities in Course#2-2019 represented as a dendrogram. Blue labels refer to higher performing students; Red labels refer to lower performing students.

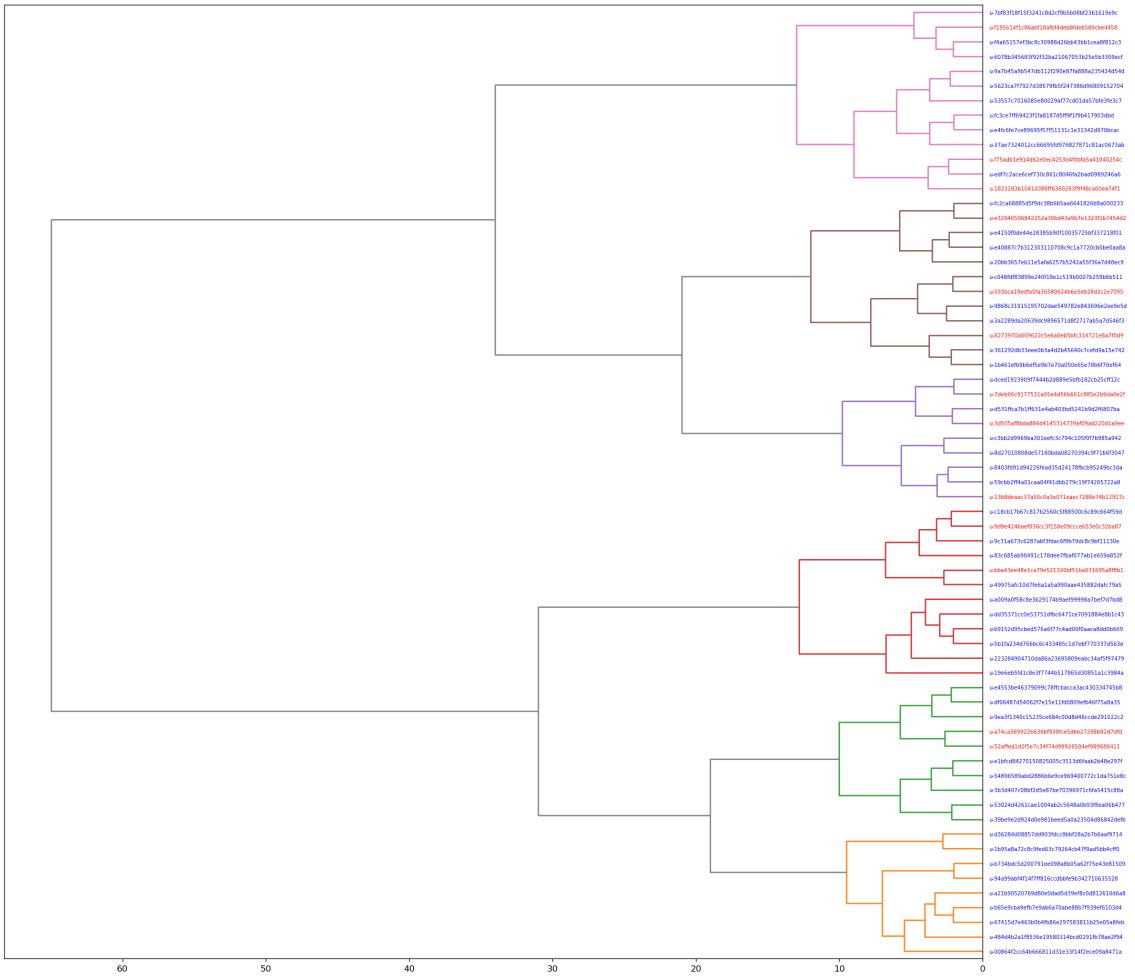


Figure B.6: Detected communities in Course#2-2020 represented as a dendrogram. Blue labels refer to higher performing students; Red labels refer to lower performing students.