Analysing students' interaction sequences on Moodle to predict academic performance

Andreia Cunha[a], Álvaro Figueira[a]

[a]Faculty of Sciences, University of Porto, Rua do Campo Alegre, 4169-007 Porto, Portugal

Abstract

As e-Learning systems have become gradually prevalent, forcing a (sometimes needed) physical distance between lecturers and their students, new methods need to emerge to fill this enlarging gap. Educators need, more than ever, systems capable of warning them (and the students) of situations that might create future problems for the learning process. The capacity to give and get feedback is naturally the best way to overcome this problem. However, in e-learning contexts, with dozens or hundreds of students, the solution becomes less simple. In this work we propose a system capable of continuously giving feedback on the performance of the students based on the interaction sequences they undertake with the LMS. This work innovates in what concerns the sequences of activity accesses together with the computation of the duration of these online learning activities, which are then encoded and fed into machine learning algorithms. We used a longitudinal experiment from five academic years. From our set of classifiers, the Random Forest obtained the best results for preventing low grades, with an accuracy of nearly 87%.

Keywords

Student grade prediction, Interaction sequences, Machine learning, Moodle logs

1. Introduction

The current pandemic situation changed how students throughout the world are learning. With education leaving the classrooms and entering students' homes, e-learning platforms such as Moodle have become popular not only as mere content distribution platforms, but as virtual spaces where students can interact with each other and their professors. However, with the transition to digital learning, face-to-face interactions have been replaced with e-mails and video conferencing. The loss of these kinds of interactions can lead to professors not having a full understanding of their students' performance, and students might not feel as motivated, resulting in lower grades or even in a complete lack of interest in the course. With these constraints, learning analytics are relevant as the educators are reduced in their capability of continuous feedback that is helpful in their activities. As such, we aim to provide a platform to ease the burden on the educators as to whether a student is performing well in their course. With the introduction of a machine learning and automated statistical analysis tool to the educators, we are in essence aiming to provide them with a smart learning environment focused on a particular online educational content distribution platform. Our research focus on Moodle as the LMS platform, due to its usage in our institution, and consequently to the ease of access

LA4SLE@EC-TEL 2021: Learning Analytics for Smart Learning Environments, September 21, 2021, Bolzano, Italy

afdcunha42@gmail.com (A. Cunha); arf@dcc.fc.up.pt (Á. Figueira)
0000-0002-0507-7504 (Á. Figueira)

CEUR Workshop Proceedings (CEUR-WS.org)

Table 1
Fields on the Moodle Log

Pertaining	Fields	Example
Activity	Event context Component	Course: Technical Communication System
Student	Student's Name Origin IP Address	AName web 127.0.0.1
Interaction	Date and Time Affected user Description Event name	4/Jul/16, 15:43 - (if not applicable) The user with id 'x' viewed the course with id 'y'. Course viewed

to the relevant data. Moodle stores user interaction information in a log file in each course. The fields in the log files are shown in Table 1. The collected information allows for a wider capacity to help educators in understanding the areas their students have the most difficulties in. Moodle allows for the distribution of Resources and Activities by the educators. Resources are defined as static material, used for exposition or consultation, while activities are defined as any item that requires user interaction, such as tests, forums or work submissions.

Thus, a tool capable of analysing the students' interactions with the platforms and comparing it to a perceived optimal access pattern might provide insights to both professors and students about their potential shortcomings. Such a tool should be able to adapt itself to any course and be able to warn both professors and students should the latter's actions result in them failing the course. Ideally, the tool should also be explainable so that the people involved can understand why it raised the alerts. The optimal medium for this tool's analysis is the activity logs of the platforms in use, since they contain both time, student and activity information. It also stands to reason that an educator may design their courses with an optimal track (the order in which resources are accessed) in mind. Comparing this track to a student's activity records allows for a measurement on how distant the student is from the intended course consumption.

2. Background

Other works in this field also attempt to create a tool such as the one we envision, with the more relevant being: Ademi et al.[1] used Moodle logs for a single course relating to two academic years, 2016/2017 and 2017/2018, totalling over 326000 log entries. They filtered the data to remove non-Latin characters, as well as removing entries not corresponding to students. The classification of data was carried out with J48 decision trees using 2, 3, and 5 classes; Bayesian networks with 2 classes; and Support Vector Machines with 2 classes; with each class representing different ranges of grades. The best performing algorithm was the J48 Decision Tree with an accuracy of 89.32%. Yang et al.[2] used data from a course from the University of Tartu, Estonia, involving 242 students. Their goal was to predict if a student would pass or fail the class based on homework grades and procrastination metrics. To classify the data, they used k-means clustering for feature extraction and classified on the following algorithm: L-

SVM, R-SVM, Gaussian processes, Decision Trees, Random Forests, Artificial Neural Networks, AdaBoost, and Naïve Bayes, of which the best performing was L-SVM with an accuracy of 84.6%. Hashim et al[3] used data from bachelor study programmes of the College of Computer Science and Information Technology, University of Basra, years 2017–2018, 2018–2019. Their aim was to predict student performance on final examinations. The data was cleaned via the removal of entries with empty fields. Each entry represents a student of which after cleaning 499 exist. The used classification algorithm were: Decision Tree, Naïve Bayes, Logistic Regression, Support Vector Machine, K-Nearest Neighbour, Sequential Minimal Optimisation and Neural Network, with the best performing being logistic regression with 88.8% in predicting failing students.

In a previous work [4], the authors extracted from Moodle logs a set of existent and derived features that allowed them to predict a student's academic grade. The system was based on a three-scale classification (low grades, margin grades, high grades). Some of the features are based on the time a student spends when interacting with a given activity or resource. While Moodle registers the time a particular page or activity has been accessed, it does not register when the user has left the page. Therefore, the length of any given session is computed by the authors using an algorithm that takes into account the start time of the next activity, but also a set of heuristics that set time activity thresholds. These times are then used to analyse student activity across ten different kinds of activity, including slide decks, tests, group formation, and workshop activities. Authors then use machine learning algorithms to classify learning paths based on similarities between students based on a multidimensional n-space where n is the number of different resources or activities taken during the course and the time spent is used as the 'intensity' in the respective axis in that n-dimensional space. This work achieved 82% accuracy in the prediction of final grades at the end of the semester for low-grade students, as well as 67% (for the same class) up to the first third of the of the semester[4].

3. Our Proposal

This section describes a paradigm shift from the classification model such that the predictions can be: 1) extended to any number of resources/activities being undertaken during a course in an LMS; 2) the computation of the time taken by each student in each activity is enhanced and fine-tuned, 3) and, especially, we revise how to compare the optimal track with the student's path. In our perspective the time a student spends in an activity or in a resource is also important, but what we consider to be fundamental is the order by which the resources and activities are accessed, which is not considered in [4]. In other words, if we were to sum all times of the same time of activity much information regarding sequence of accesses (namely, coherent access paths), returns to previous resources (for example, leading to difficulties in previous topics of subjects) would offer information that can be much more rich and informative to the system and ultimately to the instructor to mitigate those problems.

3.1. Automatic data anonymization and filtering

Moodle's logs contain a number of identifying fields, such as students' full names, unique identifying IDs and IP addresses. IP addresses are discarded. Names and numeric IDs are used to generate and are replaced with SHA-256-based hashes. The logs can also contain entries pertaining to professors and other faculty staff. It is possible to filter which kinds of users appear in the anonymized logs: students, professors, and admins. This distinction is based on blacklists for professors, educators, assistants and administrators, meaning if a name is not present on any of those lists, then it is categorised as a student.

3.2. Activity/Resource extraction

In order to achieve better generalisation capabilities, our system adapts itself to accommodate all activities provided via Moodle, so that our system does not need to be tailored for each course. For this purpose, the original logs are traversed to find occurrences of existing activities, prefix them with the relevant course identifier and academic year, as well as assign it a single letter to be used in activity strings if it's deemed to be relevant for analysis. An example of the outcome of this process is shown in Table 2. Activities can be left out of this analysis by means of a blocklist which in the case below, includes the course's landing page as well as forums.

Table 2
Examples of the outcome of the activity extraction process

Activity ID	Activity Name	Identifier
DPI1001_1920_76792	File: Presentation Lecture	A
DPI1001_1920_78330	Test: Test 1 (Thursday)	T1
DPI1001_1920_78728	File: Class02	B
DPI1001_1920_87716	File: Statement (PDF)	E
DPI1001_1920_87717	Test: Test 2 (Wednesday)	T2
DPI1001_1920_96220	Workshop: Article and Submission Evaluation	G
DPI1001_1920_108210	Group choice: Group Creation	H
DPI1001_1920_116444	Assignment: Slide Submission	K

3.3. Dealing with activity durations

While it is fair to assume that one student might navigate from one resource/activity to another in minutes or even seconds (being the differential the time spent in the former one), that is not always the case. That poses a number of issues. For example: if there is no further access from that student in the log; if there is further activity but only on another day, or simply if it would be unreasonable to assume that the student spend that much time on that precise activity. These situations need to rely on a threshold, after which it is can be assumed the student has left the activity. Our approach differs from others in the sense that we do not rely on a pre-established threshold limit equal to all Moodle activities. To handle these issues, as an initial step we take the same approach, and an initial set of times, considered to be reasonable thresholds, are defined for each activity. But then, for each entry we check the activity duration in the data. The data distribution and outliers are considered for extrapolation against our established thresholds and, should the duration be greater than the thresholds, flag them as "doubtful". The number of interactions that were flagged as doubtful in the our test data (a "Technical Communication" course) across five academic years were: 11692 in 2015/16 (37.67% of total); 9193 in 2016/17 (35.14% of total), 7689 in 2017/18 (36.64% of total), 4187 in 2018/19 (23.11% of total), and 13079 in 2019/20 (36.75% of total) which makes them fairly consistent along the years. Although one third of the duration times raises concerns, it is interesting to have a system that can assure (up to a good degree) that the duration session of two-thirds of the activities are correctly computed.

3.4. Data imputation for missing and doubtful durations

Once doubtful duration times are identified, or if they are missing, they are then handled via the imputation of new values that are taken to be a reasonable approximation of a typical access duration for the particular activity. The handling of these doubtful times is their replacement with a more reasonable value depends on the current distribution of the non doubtful duration values. The average duration and the k-nearest neighbours are the most used methods, which depend on the size of the sample for that particular activity or resource [5]. For example, this means that if the time spent in Resource A by Student X is doubtful, then the system verifies all the times available for resource A, but also how the use of resources by all students compare to the use of resources by student X.

3.5. Creation of activity strings and feature generation

The authors in [4] also create activity strings to describe access to resources and activities, however, they condense the times to be the sum of the times the user spent in each particular activity, and thus the order of activity accesses is not preserved, resulting in a huge loss of information. In this work we opt for a two-branch approach: i) we generate activity strings with no time information but preserving the access order. This will allow us to directly form and compare patterns of activity between a given student and one that attained a good grade in past years. The choice to not include time information was made on the premise that it might not accurately represent a student's effort, who for example might have accessed a slide deck with the purpose of downloading it for offline study; ii) we still have the activity times to train a machine learning classifier to help us prevent and mitigate future problematic low grades.

The student activity strings are compared to a string we consider to be optimal, reflecting the way that the educator considers that the course's information should be 'consumed'. The optimal path will naturally vary by course and by year, and is obtained by asking the educator running the tool to input it. It should be noted that what we call the optimal path is not optimal in a pedagogical sense, because different students will have different educational needs so what can be considered optimal for one student may not be optimal for another. However, we consider it to be 'scientifically optimal' because this provides a single point of comparison based on the educator's project. Also, it is preferable to be warned that a student is in danger of failing when they end up passing the course (false positives) than not receiving any warnings because the system considers that a student will pass the course, then end up failing it (false negatives). To keep the generated strings to reasonable lengths, only activities deemed relevant (and not in a blocklist) are used for generating these strings. These student activity strings are space-separated single characters, or if the activity is a test, the 'Tn' combination counts as a single character. An example of such string containing 5 activities is: "A T1 B A T2". This representation allows us to extract bi- and tri-grams, which are sequences of two or three consecutive words, which in this case represent online activities (or accessed resources). The presence of bi-grams in this context means that there is a pair of activities that are performed in an order, in an analogous way for tri-grams there is an order between three activities. The motivation is that there are coherent sequences to access pedagogic materials. For example, accesses to handouts before undertaking quizzes; reading posts before posting; obtaining templates before submitting papers, etc. These strings allow us to generate new features to complement the features already present in the previous work. These being: a) The String Length: the total activity count on a given access string. Note: it is *not* the character count of that string, but the activity count; b) Distance: the Levenshtein distance of a student's access string to the optimal access string; c) Similarity: a score of the best fitting partial substring regarding the optimal string; d) number of 2- and 3-grams (unique) that are common to the optimal string.

4. Experimental settings and preliminary results

4.1. Data balancing

First we divide the student grades (originally in a scale from 0 up to 20) into three categories: **A** (ranging from 0 to 8), **B** (9 to 11), **C** (12 to 20). Due to the imbalanced nature of the data, each classifier was tested using different sampling techniques to address this imbalance, such as oversampling, undersampling, and class weights. The dataset of initially 150k interactions was reduced to 93k (just student interactions). For the testing we had 625 rows, each agglomerating all features and corresponding to a single student attending the course from school years 2015/16 up to 2018/19. They were split 70/30 for training and testing sets, meaning 437 rows for training and 188 for testing before any class balancing effort. The training set consists of 83 samples (class A), 81 samples (class B) and 273 samples (class C). The testing set consists of 38 samples of class A, 33 samples of class B and 117 samples of class C.

4.2. Classifiers

Three types of classification algorithms were tested: Decision Trees (DT), random forests (RF), and k-Nearest Neighbours (KNN). DT tests were carried out with various maximum depths for these trees: 6, 7, and 8. The best performing DT had a depth of 8 and utilised class weights. For RF, tests were carried out in a grid with different numbers of trees (100, 150, 200), and maximum depths (6, 7, 8). The best performing RF used maximum depth 8, with 200 trees, and utilised oversampling. For KNN, tests were carried out with 2, 4, 8, 10, and 20 neighbours. Together with oversampling, lower numbers of neighbours produced better results in this category as opposed to the higher numbers tested. The best KNN run with neighbours=2. The results of each of the classifiers can be seen in Table 3. Considering the work [4] as the baseline, we globally improved the accuracy by 5% and the precision by 11% (up to 97%) in the focus class A.

5. Discussion

The reported results indicate that the classifiers have a good capability to separate the classes chosen, this is apparent when analysing the achieved metrics. Whilst the data is quite noisy, with each individual feature not being a particularly good indicator on all of their possible

Table 3
Results for the best-performing classifiers

Classifier	Accuracy	Precision			Recall			F1 score			Support		
		A	B	C	A	B	C	A	B	C	A	B	C
DT	0.72	0.81	0.72	0.85	0.59	0.81	0.76	0.69	0.76	0.70	37	32	34
RF	0.87	**0.97**	0.82	0.82	**0.90**	0.83	0.87	**0.94**	0.82	0.84	121	117	113
KNN	0.80	0.82	0.77	0.81	**0.90**	0.83	0.87	**0.94**	0.82	0.84	121	117	113

scores, the interactions between the features seem to provide the classifiers with sufficient data to achieve good results. A potential issue with the system is that the optimal string for one course on a given school year may not yet be applicable to other courses or even different instances of the same course. Depending on how long, or short, those strings end up being, it may result in individual features being too clustered around a small range of values being incapable of providing significant discriminatory performance on the classes.

6. Conclusions and Future Work

The results achieved in the different classifiers show promising accuracy with the best being the random forests classifier. These results indicate there is also a margin for more fine tuning of the classifiers and the narrowing down of the semi-arbitrary characteristics of some of the features, such as the ideal string, which should lead in future research to better results. Furthermore, the ability to choose different times for each activity to be considered doubtful, as well as the ability to tune the optimal string to the lecturer's preferences, allow for customization of the system and adapt to different circumstances, courses, and also evaluation schemes.

References

[1] N. Ademi, S. Loshkovska, S. Kalajdziski, Prediction of student success through analysis of moodle logs: Case study, in: Int. Conf. on ICT Innovations, Springer, 2019, pp. 27–40.

[2] Y. Yang, D. Hooshyar, M. Pedaste, M. Wang, Y.-M. Huang, H. Lim, Predicting course achievement of university students based on their procrastination behaviour on moodle, Soft Computing 24 (2020) 18777–18793.

[3] A. S. Hashim, W. A. Awadh, A. K. Hamoud, Student performance prediction model based on supervised machine learning algorithms, in: Materials Science and Engineering, volume 928, IOP Publishing, 2020, p. 032019.

[4] B. Cabral, Á. Figueira, A machine learning model to early detect low performing students from lms logged interactions, in: EMENA-ISTL Information Systems and Technologies to Support Learning, Springer, 2019, pp. 145–154.

[5] A. Aleryani, W. Wang, B. Iglesia, Dealing with missing data and uncertainty in the context of data mining, in: Hybrid Artificial Intelligence Systems, Springer, 2018, pp. 289–301.