Student Behaviour Models for a University LMS

Giulio Biondi[1], Valentina Franzoni[1,2(✉)], Alessio Mancinelli[1], and Alfredo Milani[1]

[1] Department of Mathematics and Computer Science, University of Perugia, Perugia, Italy
{giulio.biondi,alfredo.milani}@unipg.it, valentina.franzoni@dmi.unipg.it, alessio.mancinelli@studenti.unipg.it
[2] Department of Computer Science, Hong Kong Baptist University, Kowloon, Hong Kong, China

Abstract. Learning Management Systems (LMSs) enable teachers and educational institutions to manage the organization of the courses offered and deliver courses in blended form, with LMSs offering support to in-person teaching, or fully online. LMSs, despite having been used for a long time, saw a dramatic increase in usage due to the Covid-19 pandemic; the purpose of this study is the analysis of student behaviour within the Moodle platform, by exploiting the user interaction logs as recorded by the platform itself. Two models are proposed to predict the final outcome of students' exams based on their behaviour within the platform. The first model consists of a support vector machine, while the second model consists of an artificial neural network; both models were tested on two real-world data sets, delivering outstanding results in terms of accuracy, above 90% for some of the tested configurations.

Keywords: e-learning · Behavioural models · Academic learning

1 Introduction

The term *user profiling* indicates the process of collecting personal user data aimed at creating an abstract image, which represents the cognitive and intellectual abilities, preferences and interactions that the user has with the system, which can be used to predict the user behaviour and support, facilitate or guide its activity in the interaction with the system itself. All this information will then be stored and processed dynamically, as user behaviour can still vary over time; this structured knowledge and the protocols to infer from and interact with it represent the *user model*.

This work has been supported by MIUR PRIN PHRAME Project, and EmoRe.

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022
O. Gervasi et al. (Eds.): ICCSA 2022 Workshops, LNCS 13379, pp. 33–43, 2022.
https://doi.org/10.1007/978-3-031-10545-6_3

1.1 User Behaviour Data Sources

Information about users can be obtained in two main ways: the first one is to explicitly request from users information about themselves before they start interacting with the system. This can be achieved by an interview conducted during the registration phase, where specific data can be collected and allowed to be modified later by the user. While in principle this is a straightforward way to collect data quickly, on the other hand building a user model based on this information, exposes it to the risk of quickly becoming obsolete and most of all of not being fully objective and trustworthy. Reliability would depend entirely on the collaborative attitude of the users to provide sincere information about themselves, their attitudes, their interests, and its continuous update. It is very unlikely that further updates would happen once the initial registration process is over and, obviously, the user bias about her/himself cannot be eliminated. One advantage of this approach, however, is that it allows the user to have full control over the data that the system has access to. Another more objective approach is to allow the system itself to collect observations of user behaviour; in this case, the system monitors, as an example, the sequence of actions that the user performs to solve a given task, and therefore the information, unlike the first method, is not user provided, and it is objective. A problem in this case is that, to guarantee a high accuracy of the model without having initial data, the system has to collect a certain amount of information before being adequately performant. Therefore, it will not be able, initially, to make any prediction by facing the so called 'Cold start' [1] issue. However, when the amount of information becomes sufficient, the models will start to provide useful information and can then be adjusted automatically, allowing the model to adapt to user behaviour evolution [14].

1.2 Adaptable and Adaptive Systems

Adaptation refers to the ability of the system to react differently to each user by creating a single model of each of them, and adapting its response over the time when user behaviour evolves.

Systems with adaptation can be divided into two types based on how they use the user model: *adaptable systems* and *adaptive systems*. In the first class of systems, it is the user himself that can modify the aspect and the functionality of the system going to actively select the possible options. In the second case instead, the adaptation is executed autonomously by the system in a dynamic way, using the currently created user model [3,12,16]. Adaptability can be achieved, for instance, by manipulating the order in which content is shown to a user, thus modifying the navigation itself. One approach based on adaptive system is collaborative filtering, where information about a user is compared with that of others interaction with the same system. That is, if the characteristics of the current user match those of another, the system can make assumptions about the current user assuming that he or she has similar characteristics in areas where its model lacks enough data, by making changes based on them. Each

user has his or her own characteristics and interests, as well as varying levels of familiarity with the subject under consideration; a system that knows nothing about its users would behave in exactly the same way for each of them. It is correct to consider, for instance, even the limitations of the users, such as disability, to allow the system to assist them. Considering the context of adaptive E-learning systems [17], a user model can be defined as an abstract image of the user within the system [6]. This image must contain information inherent to its domain of knowledge, such as progress in study, goals, and interests. Based on this model, the system will then be able to adapt its behaviours, providing, as an example, more detailed material for users with higher proficiency in a given subject. These elements are well summarized by the definition of an adaptive e-learning system according to Stoyanov and Kirschner:

"An adaptive e-learning system is an interactive system that personalizes and adapts e-learning content, pedagogical models, and interactions between participants in the environment to meet the individual needs and preferences of users if and when they arise."

1.3 Learning Management Systems and Moodle

A Learning Management System is typically a web based application platform that simplifies the management, distribution and analysis of an organization's online training program. These platforms allow users, typically students and instructors, to enrol, access and hold online educational activities corresponding to their role, by giving them the ability to access them at any time. Given these features, for instance, LMSs are used as verification tools for a group of company employees to assess their competence level and productivity, or for a class of university students for conducting ongoing evaluation tests to assess their knowledge level. It is particularly important the role of high-level users like administrators, whose task is to create courses and training plans, track the progress of those who use the LMS and upload contents within the platform, and instructors, which can monitor at any time the progress made by specific learners in a specific course [9]. Depending on the LMS used, different information is stored for each user, such as the total time they spend in a single course, the number of times they access the platform or performance automatically obtained in a given test (e.g. a multiple answer quiz), or assessed by the instructor for a given assignment. To promote the 'social learning' modality, an LMS can include features that support collaboration, such as forums and wikis where it is possible to create discussion groups or develop specific content on a specific topic collaboratively. In order to increase user engagement, gamification tools can also be used, through the installation of plugins, which will eventually reward the user through a system of rewards based on the score obtained. Corporate training is not the only application of LMS as they are also used within educational institutions, finding applications in schools and universities. Moodle, for instance, is an LMS distributed free of charge under an open source license, giving organizations the possibility to access the source code, thus enabling them to create numerous additional plugins aimed at adding functionality to improve the user

experience of students as well as instructors. The learning style that Moodle follows is based on the interaction that the user has both with the learning material [5] and, most importantly, with other learners of the same by various forms of collaboration [11,13]. This system offers enormous advantages to teachers, since, after creating the course, it allows them to publish the material of the lessons from time to time, and also insert specific quizzes to check the student's learning progress. The platform allows the instructor, for instance, to keep track of which student has been most/least active throughout the course [2], since the system stores the logs (i.e. all the events that a user performs) in a relational database [8,15], the instructor can therefore plan learner specific activity/actions such as deepening a certain topic or repeating difficult topics. Moodle LMS stores information about user activities on more than 145 interrelated tables [4]; in this paper, we focus on the data of table mdl_logstore_standard_log: it collects all the registration of user activities and information about the events taking place in a given course, also including activities carried out within forums or wikis.

In this paper we will describe the architecture of a system using machine learning techniques to make predictions on *final exams outcomes* of university students; the system uses logs from a Moodle platform for building a meaningful user model. In Sect. 2 the data preprocessing, filtering and feature selection phases to build the data set used in the experiments are described. In Sect. 3 the machine learning architecture, the experiments held and the obtained results are presented and discussed. Conclusions are finally drawn in Sect. 4.

2 Data Preprocessing, Features Selection and Datasets

The data we considered describe the history of a university course supported by a Moodle e-learning platform in mixed modality, i.e. lessons held in physical classes, but supported by material distributed online and tests, quizzes, and assignments submitted online. The events took place over a time span of 24 months, or 101 weeks, from the actual course creation date in the LMS to the effective course ending date; a few weeks were excluded because no user events were recorded. Figure 1 and Fig. 2 show the distribution of the events, respectively, per month and per week across the selected time span.

The mdl_logstore_standard_log Moodle records for each event 21 different attributes; for the purpose of this work, four were considered:

- *userid*
- *eventname*
- *timecreated*
- *objectid*

During the first weeks, the system recorded only a few events; it is apparent that from March, i.e. the course starting month onwards, they become much more frequent with peaks in the June/July 2018 and January/February 2019 periods. The scarcity of events in the beginning weeks is due to the date of Moodle course creation not necessarily corresponding to the actual start date of

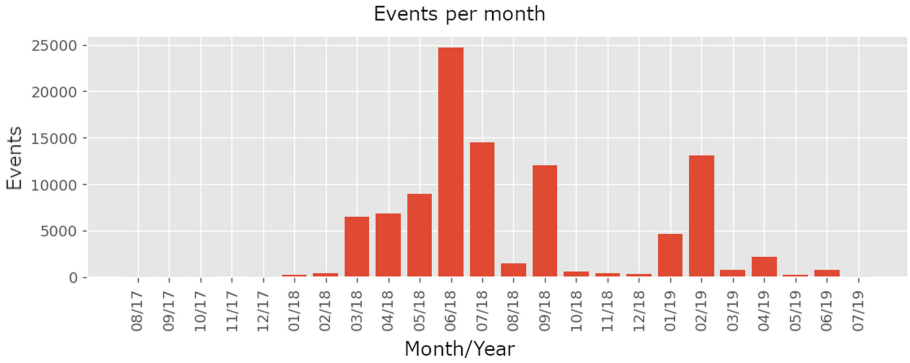

Fig. 1. Events per month

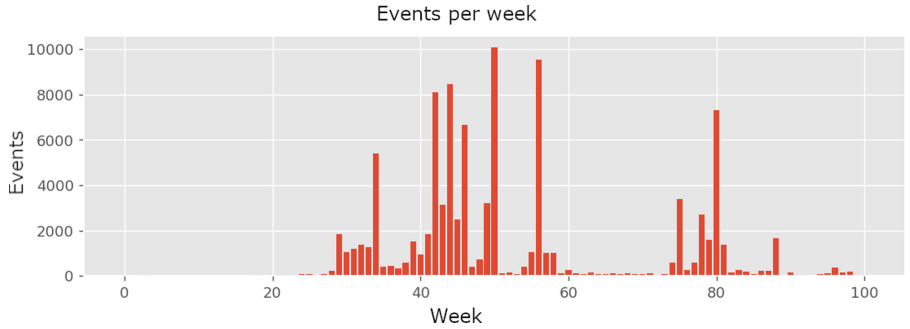

Fig. 2. Events per week

the course. The events in the first months, which constituted a tiny fraction of the total in the data set, mainly represented course enrollment events and were thus deemed not significant for the work. Therefore, 2018-01-01 was chosen as the course start date; the course ending date was set instead to April 16th 2019, the last exam session of which results were available. All of the events within the original data set were generated by 620 single users, including administrators and system applications; filtering the users to consider students only lowered the figure to 605 and applying the date-time filtering described above, the users finally fell to 575. Of these, 368 took the exam, while 207 did not attempt any session within the analysed period. The initial total number of events by the 620 users was 99011, reduced to 93024 after applying the filter on students, and finally to 91564 by considering only those events within the course period. Considering the users who passed the exam, all the events following the successful attempt were filtered out since they were deemed not meaningful to train the models. The rest of the data set consists of students who have never passed the exam or attempted it. For such students, the date of the first event recorded since the beginning of the course was also taken into account but, as the final

date, April 16th, 2019, i.e. the last considered date, thus not filtering out any event.

April Data Set. All the features extracted, as described in Subsect. 2.1, from the 575 students were used to build the first data set, the *April data set*.

July Data Set. A second data set, the *July data set*, was built by considering all the students who took the exam within the month of July, i.e. during the first exam session; it contains data of events from 2018-01-01, i.e. the course beginning date, to 2018-07-24, i.e. the last July exam session date.

2.1 Feature Selection

The following per-user features were selected and extracted from data on a time-based interval to build the *April* and *July* data sets:

# of events Each row within the data set corresponds to an event performed by the user uniquely identified in the 'userid' column; therefore, counting the occurrences of each user-id within the data set yields the total events.

# of course accesses The event '$\backslash core \backslash event \backslash course_viewed$' logs the accesses to the course; the total number of accesses per user was counted. Furthermore, a key aspect linked to the total accesses is their temporal distribution over the different weeks and months of the course. The assumption is that users who accessed the course from the earliest days, on a recurrent basis, will have better exams outcome on average and should thus be assigned more credit than those who waited for the exam sessions.

# of viewed discussions The event '$\backslash mod_forum \backslash event \backslash discussion_viewed$' records discussions' views by users on the course forum, which is typically used to communicate course information to students, and stores the reference to the discussion in the *objectId* attribute. The number of different discussions viewed by each user is counted.

# of viewed resources similarly to obtaining the number of discussions; in this case the event considered is: '$\backslash mod_resource \backslash event \backslash course_module_viewed$'

final exam outcome Indicates the final outcome of each user's exam, i.e. the variable that the models will predict.

3 Experiments Design and Results

In this section, the machine learning architecture, the experiments held and the obtained results are presented and discussed.

3.1 Experiments Design and Architecture

In this work two ML classifiers have been employed, i.e. Artificial Neural Networks (ANNs) [10] and Support Vector Machines (SVM) [7].

K-fold Validation. After preliminary experiments which showed a tendency to overfit with both ANN and SVM, a K-fold cross validation technique was employed to divide the data set into $k = 5$ partitions of equal size. In cross-validation, a system is trained on $k-1$ partitions and tested on the kth remaining partition; the process is repeated until each partition has been used as a test set. Considering the *April data set*, for each partition 460 of 575 users, i.e. 80% of the original data set were used as the training set and 115, i.e. the remaining 20 as the testing set. For the *July data set*, the total number of students was 340, with 272 used as training set and 68 as test set.

SVM. Two SVM kernels were tested, i.e. linear and RBF; various parameter values were tested for each kernel to find the best performing combinations. The parameter values were selected following a *grid search* approach, exhaustively testing, in selected ranges, combinations of the parameters subject to optimization, and evaluating the performance of the classifier for each parameters combination. This approach generally guarantees good results at the price of the high computational capacity required to train from the beginning a classifier for each of the combinations tested. The ranges of values used for the choice of parameters are reported in Table 1, while the final values chosen for the experiments are reported in Table 2.

Table 1. SVM grid search ranges

Parameter	Values					
C	0.001	0.01	0.1	1	10	100
γ	0.0001	0.001	0.01	0.1	1	10

Table 2. SVM parameters

Kernel	C	γ
Linear	0.1	Not applicable
RBF	100	0.0001

ANNs. A Fully Connected Neural Network has been designed, with the following architecture: the input layer has 88 neurons (resp. 41), i.e. the number of features in the *April* data set (resp. *July*). The two hidden layers are composed respectively of 32 and 16 nodes with *relu* activation function, while the final output layer consists of a single neuron with *sigmoid* activation function. The *Adam* optimizer was used in the training process; binary cross-entropy was chosen as loss function and Accuracy as performance metric.

Evaluation Metrics. The main evaluation metric chosen for the experiments is *accuracy*.

Let define:

- TP_{Passed} = True Positives i.e., the number of correctly classified items belonging to class *Passed Exam*
- FN_{Passed} = False Negatives i.e., the number of misclassified items belonging to class *Passed Exam*
- FP_{Passed} = False Positives i.e., the number of *Failed Exam* items erroneously classified as *Passed Exam*
- TN_{Passed} =True Negative i.e., the number of correctly classified *Failed Exam* items

Accuracy is defined, for binary classification tasks, as the ratio of the correctly classified records to the total number of records:

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Other computed metrics include per-class *Precision*, *Recall*, and *F1-Score*.

3.2 Experiments Result

In this section the results of the experiments on the July and April data sets are reported; all the figures in the tables are averaged over 5 folds.

July Data Set. In Table 3(a), Table 3(b) and Table 3(c) the results for, respectively, the SVM models with Linear (SVM/L) and RBF (SVM/RBF) kernels and the ANN model are reported.

The best results are achieved by the SVM/L model, with an Accuracy of 85% in predicting the exam outcome; the other models show slightly lower performance, while still equal or above 80%.

Table 3. Average results on the July data set

Class	Precision	Recall	F-score	Accuracy	Support
(a) SVM/L model, C = 0.1					
0	82%	82%	82%	85%	28
1	87%	87%	87%		40
(b) SVM/RBF model, C = 100, γ= 0.0001					
0	78%	76%	76%	81%	28
1	83%	84%	83%		40
(c) ANN model					
0	80%	71%	74%	80%	28
1	81%	87%	84%		40

Table 4. Average results on the April data set

	Precision	Recall	F-score	Accuracy	Support
(a) SVM/L model, C = 0.1					
0	95%	97%	96%	96%	52
1	98%	96%	97%		63
(b) SVM/RBF model, C = 100, γ= 0.0001					
0	91%	88%	89%	90%	52
1	90%	92%	91%		63
(c) ANN model					
0	86%	83%	84%	89%	52
1	86%	89%	87%		63

April Data Set. In Table 4(a), Table 4(b) and Table 4(c) the results for, respectively, the SVM/L, SVM/RBF and ANN models are reported.

The figures on the April data set are higher for all the models when compared to the July data set; such results can be traced to the higher number of samples and the wider set of features, covering a longer timeframe. The lowest Accuracy score, as in the case of the July data set, is obtained by the ANN model; furthermore, similarly to the other data set, the ANN and SVM/RBF models offer comparable performance, albeit with a higher ≈90% Accuracy. The SVM/L model shows the highest figure, with a remarkable 96% Accuracy on 115 samples and comparable Precision, Recall and F-score values across the two classes. All models offer better performance on the April dataset; ANN turned out to be, in both datasets, the model that offers worse accuracy, although very high. We can see that the performance of ANN and the SVM model with RBF kernel are similar, but the latter offers better performance in both datasets. In general, however, we can conclude that all the proposed models offer a good solution to the problem, guaranteeing high accuracy in assigning an exam outcome given the behaviour of a student within the platform. The model that offers better performance for the posed problem is the SVM having linear Kernel and a value of 0.1 for the C parameter.

4 Conclusions

This research work shows and analyzes how students approach the Moodle platform, how different their behaviours can be, and how such behaviours can lead to predictable consequences in terms of passing an exam. All of the proposed models showed good performance in solving the problem at hand; in particular, the SVM model achieved very high accuracy, almost always correctly predicting the outcome of the exam of the analyzed students.

References

1. Abbakumov, D.: The solution of the "cold start problem" in e-learning. Procedia - Social Behav. Sci. **112**, 1225–1231 (2014). https://doi.org/10.1016/j.sbspro.2014.01.1287
2. Bañeres, D., Rodríguez-Gonzalez, M.E., Serra, M.: An early feedback prediction system for learners at-risk within a first-year higher education course. IEEE Trans. Learn. Technol. **12**, 249–263 (2019)
3. Beck, J.E., Woolf, B.P.: High-level student modeling with machine learning. In: Gauthier, G., Frasson, C., VanLehn, K. (eds.) ITS 2000. LNCS, vol. 1839, pp. 584–593. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-45108-0_62
4. Bognár, L., Fauszt, T.: Different learning predictors and their effects for moodle machine learning models. In: 2020 11th IEEE International Conference on Cognitive Infocommunications (CogInfoCom), pp. 000405–000410 (2020). https://doi.org/10.1109/CogInfoCom50765.2020.9237894
5. Bovo, A., Sanchez, S., Héguy, O., Duthen, Y.: Clustering moodle data as a tool for profiling students. In: 2013 Second International Conference on E-Learning and E-Technologies in Education (ICEEE), pp. 121–126 (2013). https://doi.org/10.1109/ICeLeTE.2013.6644359
6. Ciloglugil, B., Inceoglu, M.M.: User modeling for adaptive e-learning systems. In: Murgante, B., et al. (eds.) ICCSA 2012. LNCS, vol. 7335, pp. 550–561. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-31137-6_42
7. Cortes, C., Vapnik, V.: Support-vector networks. Mach. Learn. **20**(3), 273–297 (1995). https://doi.org/10.1007/BF00994018
8. Dutt, A., Ismail, M.A., Herawan, T.: A systematic review on educational data mining. IEEE Access **5**, 15991–16005 (2017). https://doi.org/10.1109/ACCESS.2017.2654247
9. Franzoni, V., Milani, A., Mengoni, P., Piccinato, F.: Artificial intelligence visual metaphors in e-learning interfaces for learning analytics. Appl. Sci. **10**(20) (2020). https://doi.org/10.3390/app10207195, https://www.mdpi.com/2076-3417/10/20/7195
10. Haykin, S.: Neural Networks: A Comprehensive Foundation. Prentice Hall PTR, Upper Saddle River (1994)
11. Iglesias-Pradas, S., Ruiz-de Azcárate, C., Agudo-Peregrina, Á.F.: Assessing the suitability of student interactions from Moodle data logs as predictors of cross-curricular competencies. Comput. Human Behav. **47**, 81–89 (2015).https://doi.org/10.1016/j.chb.2014.09.065, http://www.sciencedirect.com/science/article/pii/S0747563214005238
12. Li, N., Cohen, W., Koedinger, K., Matsuda, N.: A machine learning approach for automatic student model discovery. In: EDM 2011 - Proceedings of the 4th International Conference on Educational Data Mining, pp. 31–40 (2011)
13. Mengoni, P., Milani, A., Li, Y.: Impact of time granularity on histories binary correlation analysis. In: Misra, S., et al. (eds.) ICCSA 2019. LNCS, vol. 11620, pp. 323–335. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-24296-1_27
14. Montaner, M., López, B., de la Rosa, J.L.: A taxonomy of recommender agents on the internet. Artif. Intell. Rev. **19**(4), 285–330 (2003). https://doi.org/10.1023/A:1022850703159
15. Peña-Ayala, A.: Educational data mining: a survey and a data mining-based analysis of recent works. Expert Syst. Appl. **41**(4, Part 1), 1432–1462 (2014). https://doi.org/10.1016/j.eswa.2013.08.042, http://www.sciencedirect.com/science/article/pii/S0957417413006635

16. Sison, R., Shimura, M.: Student modeling and machine learning. Int. J. Artif. Intell. Educ. (IJAIED) **9**, 128–158 (1998). https://telearn.archives-ouvertes.fr/hal-00257111
17. Wang, Y.h., Liao, H.C.: Data mining for adaptive learning in a TESL-based e-learning system. Expert Syst. Appl. **38**(6), 6480–6485 (2011). https://doi.org/10.1016/j.eswa.2010.11.098