ISSN: 2454-9940

INTERNATIONAL JOURNAL OF APPLIED SCIENCE ENGINEERING AND MANAGEMENT

E-Mail :
editor.ijasem@gmail.com
editor@ijasem.org

www.ijasem.org

INTERNATIONAL JOURNAL OF APPLIED SCIENCE ENGINEERING AND MANAGEMENT
ISSN 2454-9940
www.ijasem.org
Vol 18, Issue 4, 2024
https://doi.org/10.5281/zenodo.14065961

A FRAMEWORK FOR STUDENT PROFILE IDENTIFICATION IN ONLINE JUDGE SYSTEMS USING EXPLAINABLE AI

[1]K. Kalyani,[2]Bathini Prashanth
[1]Assistant Professor, [2]MCA Student
Department Of MCA Student
Sree Chaitanya College of Engineering, Karimnagar

ABSTRACT

Because they provide quick and impartial evaluations of the code that students write, online judge (OJ) systems are frequently taken into consideration in programming-related courses. Based on a rubric, such an evaluation often yields a single conclusion, usually indicating whether the submission completed the task satisfactorily. Nevertheless, it would be advantageous for the student and the teacher to get more feedback regarding the task's overall progress, as such data can be considered inadequate in an educational setting. By taking into account the potential for future utilisation of the data collected by the OJ and automatically deriving feedback for both the teacher and the student, this study attempts to address this issue. More specifically, we examine the modelling of student behaviour using learning-based schemes, including Multi-Instance Learning and conventional Machine Learning formulations. Additionally, Explainable AI is being considered to give feedback that is intelligible to humans. The concept was assessed using a case study that included 2,500 entries from about 90 different students enrolled in a computer science degree course that dealt with programming. The outcomes gained support the proposal: based just on the behavioural pattern deduced from the submissions made to the OJ, the model can accurately anticipate the user outcome (passing or failing the assignment). Additionally, the proposal may pinpoint student groupings and profiles that are more likely to fail as well as other pertinent data, which ultimately provides feedback to the teacher and the student.

1. INTRODUCTION

ORIGINALLY coined by [1], the term Online Judge (OJ) denotes those systems devised for the automated evaluation and grading of programming assignments, which usually take the form of online evaluation services capable of collecting source codes, compiling them, assessing their results, and computing scores based on different criteria [2]. These automated tools have been particularly considered in two precise, yet related, scenarios [3]: (i) programming contests and competitions, and (ii) educational contexts in academic degrees. This work focuses on the latter scenario, in particular, on programming courses from Computer Science studies in higher education institutions.

INTERNATIONAL JOURNAL OF APPLIED SCIENCE ENGINEERING AND MANAGEMENT
https://doi.org/10.5281/zenodo.14065961
ISSN 2454-9940
www.ijasem.org
Vol 18, Issue 4, 2024

OJ systems are successful in the education field because they overcome the main issues associated with the manual evaluation of assignments [4]: in opposition to human grading, which is deemed as a tedious and error-prone task, these tools provide immediate corrections of the submissions regardless of the number of participants. Moreover, the competitive learning framework that these schemes entail proves to benefit the success of the learning process [5].

Despite their clear advantages, OJ systems do not provide the student nor the instructor with any feedback from the actual submission apart from whether the provided code successfully accomplished the assignment [6]. However, the information gathered by the OJ system may be further exploited to enrich the educational process by automatically extracting additional insights such as student habits or patterns of behaviour related to the success (or failure) of the task. In this regard, one may resort to the so-called Educational Data Mining (EDM), a discipline meant to infer descriptive patterns and predictions from educational settings [7]. Within this discipline, Machine Learning (ML) is reported as one of the main enabling technologies due to its power and flexibility. Some success cases can be found in the work by [8], devoted to assessing the performance of the instructor; the approach by [9], aimed at predicting student grades at an early stage; or the work by [10], focused on detecting inconsistencies in peer-review assignments. In this work, we apply EDM to automatically provide feedback about the assignments, both to the student and the instructor, in the context of OJ systems for programming courses.

When an OJ is used for grading a programming assignment, there is usually a time slot in which students can perform as many submissions as they want. The final grade of a student in the activity is typically computed from the best submission. During that time slot, data usually exploited in EDM, such as grades obtained in previous activities or course attendance [9], may not be available. Moreover, other data used to predict student performance, such as socioeconomic background or academic success in other courses [11], may not be usable from an ethical point of view due to the potential biases it would introduce.

In spite of the lack of available data, it would still be desirable to be able to detect at-risk students before the assignment deadline. Thus, aided by the use of meta-information gathered from the submission process—e.g., the number of code submission attempts or the date of the first submission—we devised an EDM approach with two types of outcomes: (i) the success probability of a new student, and (ii) the identification of different student profiles to provide feedback to both the instructor and the student thyself. Note that such pieces of information may be used not only to prevent inadequate student attitudes by providing the appropriate observations about the development of the task but also to properly

INTERNATIONAL JOURNAL OF APPLIED SCIENCE ENGINEERING AND MANAGEMENT
https://doi.org/10.5281/zenodo.14065961
ISSN 2454-9940
www.ijasem.org
Vol 18, Issue 4, 2024

adjust the difficulty of the different assignments, among other possible corrective actions towards the success of the course.

Since the set of code submissions made by a student somehow characterises the student profile to be estimated, the problem may be modelled as a Multi-Instance Learning (MIL) task [12]. This learning framework introduces the concept of bag, i.e., a set with an indeterminate number of instances that is assigned a single label [13]. MIL has been successfully considered in the EDM literature [14], as in the work by [15], which compares MIL against ML for predicting the student performance. In our case, each of these bags gathers the different code submissions made by each student, being labelled as either positive or negative depending on whether the student eventually passed the assessment by the OJ system.

Nevertheless, the fact that both ML and MIL strategies generally work in a black box manner hinders their application in this feedback-oriented context [16]. In this regard, the field of Explainable Artificial Intelligence (XAI) is gradually gaining attention to tackle such limitation by devising methodologies that allow humans to understand and interpret the decisions taken by a computational model [17]. However, while XAI has been largely studied in the ML field, this has not been the case in the MIL one [18].

Considering all the above, this work presents a method to identify student profiles in educational OJ systems with the aim of providing feedback to both the students and the instructors about the development of the task. More precisely, the proposal exclusively relies on the meta-information extracted from these OJ systems and considers a MIL framework to automatically infer these profiles together with XAI methods to provide interpretability about the estimated behaviours. In order to apply XAI to MIL problem, a novel policy for mapping the MIL representation to an ML one is proposed for the particular task at hand. The proposed methodology has been evaluated in a case of study comprising three academic years of a programming-related course with more than 2,500 submissions of two different assignments. For this, more than 20 learning-based strategies comprising ML, MIL, and MILto-ML mapping methods have been assessed and compared to prove the validity of the proposal. The results obtained show that the proposal adequately models the user profile of the students while it also provides a remarkably precise estimator of their chances to succeed or fail in the posed task solely based on the meta-information of the OJ.

The rest of the work is organised as follows: Section II reviews the related literature to contextualise the work; Section III presents the proposed methodology; Section IV introduces the case of study examined; Section V details the experimental set-up considered; Section VI shows and discusses the results obtained;

INTERNATIONAL JOURNAL OF APPLIED SCIENCE ENGINEERING AND MANAGEMENT
ISSN 2454-9940
www.ijasem.org
Vol 18, Issue 4, 2024
https://doi.org/10.5281/zenodo.14065961

Section VII summarises the insights obtained in the work; and finally, Section VIII concludes the work and outlines future research line to address.

2. EXISTING SYSTEM

The work by [19], who was the first to propose that academic computing assignments could be automatically graded, is considered the main precursor of current OJ systems. Nevertheless, their first formal definition was introduced by [1] who described them as a computer system that automatically grades programming assignments and provides some type of feedback to the students.

Regarding their practical use, the scientific literature comprises a large number of OJ proposals related, to a great extent, to academic institutions and educational environments. Some examples of such systems comprise the work by [20] with the Javaluador method for tasks in the Java programming language (it is described later in this paper), the URI system by the Universidade Regional Integrada for developing and improving general coding skills [21], the Peking University Online Judge (POJ) by [22] tailored to C++ courses, the CourseMaker one by the University of Nottingham for general programming tasks [23], the Youxue Online Judge (YOJ) [24] also for improving coding skills inspired on exercises from different programming contests, and the Sphere Online Judge (SPOJ) devised for E-Learning frameworks [25], among others.

Besides their use for educational purposes, OJ systems are also commonly considered in the context of coding competitions for solving algorithmic problems. Examples of such cases are the one used in the International Collegiate Programming Contest [26] or the UVa one considered in the Olympiads in Informatics [27].

The identification of struggling students in early course stages is deemed as a remarkably important topic in the education field as it suggests the instructor to provide additional resources to address the problem. In this sense, a large number of studies have assessed the influence of both extrinsic and intrinsic factors on the commented difficulties.

In relation to the extrinsic aspects, most of the existing literature resorts to the analysis of the socioeconomic position of the student or the marks obtained in previous courses [11]. The reader is referred to the manuscript by [28] for a thorough revision of these factors as it is out of the scope of this work. Regarding the intrinsic aspects—using information about the outcomes of the assignments carried out within a course—, the related literature comprises a large number of approaches since they typically yield considerably accurate predictions. Some representative examples include: the work by [29], which addresses this task in generic online learning platforms; that by [30] on preventive failure detection in the context of the Moodle platform; the case of [31] that estimates this information relying on information gathered from clicker tests in

INTERNATIONAL JOURNAL OF APPLIED SCIENCE ENGINEERING AND MANAGEMENT
https://doi.org/10.5281/zenodo.14065961
ISSN 2454-9940
www.ijasem.org
Vol 18, Issue 4, 2024

peer-based instruction environments; and the approach by [9], who use course attendance as a predictor of academic outcome for the academic year.

Focusing on the case of programming courses, it may be checked that the most basic, yet successful, approaches rely on hand-crafted heuristics neglecting the use of OJ systems. For instance, Error Quotient [32] together with its refined version Repeated Error Density [33] perform this assessment by resorting to the syntax errors that occur during the compilation stage. The Watwin Scoring Algorithm [34] works in a similar way, but penalises students based on the time required to fix each type of error compared to that of their peers. [35] devised a scoring mechanism that takes into account more complex interactions, such as debugging or modifying syntactically correct code. A last example is the one by [36] that identifies at-risk students by means of a linear regression approach based on compilation errors and other indicators.

Disadvantages

• The complexity of data: Most of the existing machine learning models must be able to accurately interpret large and complex datasets to judge the Student profiles.

• Data availability: Most machine learning models require large amounts of data to create accurate predictions. If data is unavailable in sufficient quantities, then model accuracy may suffer.

• Incorrect labeling: The existing machine learning models are only as accurate as the data trained using the input dataset. If the data has been incorrectly labeled, the model cannot make accurate predictions.

3. PROPOSED SYSTEM

Considering all the above, this work presents a method to identify student profiles in educational OJ systems with the aim of providing feedback to both the students and the instructors about the development of the task. More precisely, the proposal exclusively relies on the meta-information extracted from these OJ systems and considers a MIL framework to automatically infer these profiles together with XAI methods to provide interpretability about the estimated behaviours. In order to apply XAI to MIL problem, a novel policy for mapping the MIL representation to an ML one is proposed for the particular task at hand. The proposed methodology has been evaluated in a case of study comprising three academic years of a programming-related course with more than 2,500 submissions of two different assignments. For this, more than 20 learning-based strategies comprising ML, MIL, and MILto-ML mapping methods have been assessed and compared to prove the validity of the proposal. The results obtained show that the proposal adequately models the user profile of the students while it also provides a remarkably precise estimator of their chances to succeed or fail in the posed task solely based on the meta-information of the OJ.

Advantages

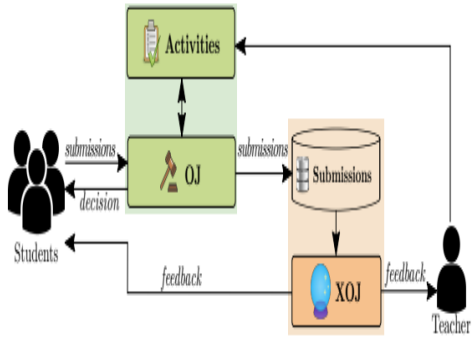
(i) transparency methods, which represent the ones that directly

INTERNATIONAL JOURNAL OF APPLIED SCIENCE ENGINEERING AND MANAGEMENT
https://doi.org/10.5281/zenodo.14065961
ISSN 2454-9940
www.ijasem.org
Vol 18, Issue 4, 2024

convey the workings of the model; and

(ii) (ii) post-hoc explanations, which attempt to provide justifications about the reason why the model arrived at its predictions. This work frames on the latter case since, oppositely to transparency-based approaches, they avoid the need for individually adapting each learning-based model considered for the particular task at hand.

4. SYSTEM ARCHITECTURE



5. ALGORITHIM

Gradient boosting

Gradient boosting is a machine learning technique used in regression and classification tasks, among others. It gives a prediction model in the form of an ensemble of weak prediction models, which are typically decision trees.[1][2] When a decision tree is the weak learner, the resulting algorithm is called gradient-boosted trees; it usually outperforms random forest.A gradient-boosted trees model is built in a stage-wise fashion as in other boosting methods, but it generalizes the other methods by allowing optimization of an arbitrary differentiable loss function.

Logistic regression Classifiers

Logistic regression analysis studies the association between a categorical dependent variable and a set of independent (explanatory) variables. The name logistic regression is used when the dependent variable has only two values, such as 0 and 1 or Yes and No. The name multinomial logistic regression is usually reserved for the case when the dependent variable has three or more unique values, such as Married, Single, Divorced, or Widowed. Although the type of data used for the dependent variable is different from that of multiple regression, the practical use of the procedure is similar.

Logistic regression competes with discriminant analysis as a method for analyzing categorical-response variables. Many statisticians feel that logistic regression is more versatile and better suited for modeling most situations than is discriminant analysis. This is because logistic regression does not assume that the independent variables are normally distributed, as discriminant analysis does.

This program computes binary logistic regression and multinomial logistic regression on both numeric and categorical independent variables. It reports on the regression equation as well as the goodness

INTERNATIONAL JOURNAL OF APPLIED SCIENCE ENGINEERING AND MANAGEMENT
https://doi.org/10.5281/zenodo.14065961
ISSN 2454-9940
www.ijasem.org
Vol 18, Issue 4, 2024

of fit, odds ratios, confidence limits, likelihood, and deviance. It performs a comprehensive residual analysis including diagnostic residual reports and plots. It can perform an independent variable subset selection search, looking for the best regression model with the fewest independent variables. It provides confidence intervals on predicted values and provides ROC curves to help determine the best cutoff point for classification. It allows you to validate your results by automatically classifying rows that are not used during the analysis.

Naïve Bayes

The naive bayes approach is a supervised learning method which is based on a simplistic hypothesis: it assumes that the presence (or absence) of a particular feature of a class is unrelated to the presence (or absence) of any other feature .
Yet, despite this, it appears robust and efficient. Its performance is comparable to other supervised learning techniques. Various reasons have been advanced in the literature. In this tutorial, we highlight an explanation based on the representation bias. The naive bayes classifier is a linear classifier, as well as linear discriminant analysis, logistic regression or linear SVM (support vector machine). The difference lies on the method of estimating the parameters of the classifier (the learning bias).

While the Naive Bayes classifier is widely used in the research world, it is not widespread among practitioners which want to obtain usable results. On the one hand, the researchers found especially it is very easy to program and implement it, its parameters are easy to estimate, learning is very fast even on very large databases, its accuracy is reasonably good in comparison to the other approaches. On the other hand, the final users do not obtain a model easy to interpret and deploy, they does not understand the interest of such a technique.
Thus, we introduce in a new presentation of the results of the learning process. The classifier is easier to understand, and its deployment is also made easier. In the first part of this tutorial, we present some theoretical aspects of the naive bayes classifier. Then, we implement the approach on a dataset with Tanagra. We compare the obtained results (the parameters of the model) to those obtained with other linear approaches such as the logistic regression, the linear discriminant analysis and the linear SVM. We note that the results are highly consistent. This largely explains the good performance of the method in comparison to others. In the second part, we use various tools on the same dataset (**Weka 3.6.0**, **R 2.9.2**, **Knime 2.1.1**, **Orange 2.0b** and **RapidMiner 4.6.0)**. We try above all to understand the obtained results.

Random Forest

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time. For

INTERNATIONAL JOURNAL OF APPLIED SCIENCE ENGINEERING AND MANAGEMENT
https://doi.org/10.5281/zenodo.14065961
ISSN 2454-9940
www.ijasem.org
Vol 18, Issue 4, 2024

classification tasks, the output of the random forest is the class selected by most trees. For regression tasks, the mean or average prediction of the individual trees is returned. Random decision forests correct for decision trees' habit of overfitting to their training set. Random forests generally outperform decision trees, but their accuracy is lower than gradient boosted trees. However, data characteristics can affect their performance.

The first algorithm for random decision forests was created in 1995 by Tin Kam Ho[1] using the random subspace method, which, in Ho's formulation, is a way to implement the "stochastic discrimination" approach to classification proposed by Eugene Kleinberg.

An extension of the algorithm was developed by Leo Breiman and Adele Cutler, who registered "Random Forests" as a trademark in 2006 (as of 2019, owned by Minitab, Inc.).The extension combines Breiman's "bagging" idea and random selection of features, introduced first by Ho[1] and later independently by Amit and Geman[13] in order to construct a collection of decision trees with controlled variance.

Random forests are frequently used as "blackbox" models in businesses, as they generate reasonable predictions across a wide range of data while requiring little configuration.

SVM

In classification tasks a discriminant machine learning technique aims at finding, based on an independent and identically distributed (iid) training dataset, a discriminant function that can correctly predict labels for newly acquired instances. Unlike generative machine learning approaches, which require computations of conditional probability distributions, a discriminant classification function takes a data point x and assigns it to one of the different classes that are a part of the classification task. Less powerful than generative approaches, which are mostly used when prediction involves outlier detection, discriminant approaches require fewer computational resources and less training data, especially for a multidimensional feature space and when only posterior probabilities are needed. From a geometric perspective, learning a classifier is equivalent to finding the equation for a multidimensional surface that best separates the different classes in the feature space.

SVM is a discriminant technique, and, because it solves the convex optimization problem analytically, it always returns the same optimal hyperplane parameter—in contrast to genetic algorithms (GAs) or perceptrons, both of which are widely used for classification in machine learning. For perceptrons, solutions are highly dependent on the initialization and termination criteria. For a specific kernel that transforms the data from the input space to the feature space, training returns uniquely defined SVM model parameters for a given training set, whereas the perceptron and GA classifier models are different each time training is initialized. The aim of GAs and perceptrons is only to minimize error during training,

INTERNATIONAL JOURNAL OF APPLIED SCIENCE ENGINEERING AND MANAGEMENT
https://doi.org/10.5281/zenodo.14065961
ISSN 2454-9940
www.ijasem.org
Vol 18, Issue 4, 2024

which will translate into several hyperplanes' meeting this requirement.

6. Modules

Service Provider

In this module, the Service Provider has to login by using valid user name and password. After login successful he can do some operations such as Browse Students Datasets and Train & Test Data Sets, View Trained and Tested Accuracy in Bar Chart, View Trained and Tested Accuracy Results, View Prediction Of Online Student's Profile judgement, View Online Student's Profile judgement Ratio, Download Predicted Data Sets, View Online Student's Profile judgement Type Ratio Results, View All Remote Users.

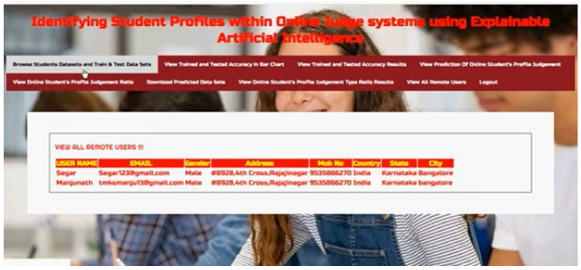
View and Authorize Users

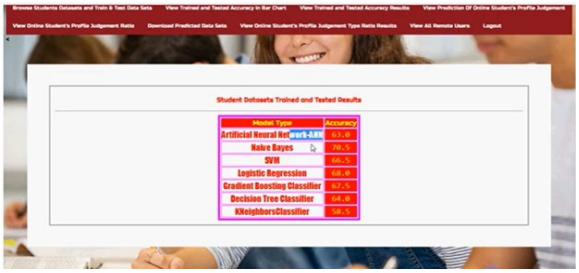
In this module, the admin can view the list of users who all registered. In this, the admin can view the user's details such as, user name, email, address and admin authorizes the users.

Remote User

In this module, there are n numbers of users are present. User should register before doing any operations. Once user registers, their details will be stored to the database. After registration successful, he has to login by using authorized user name and password. Once Login is successful user will do some operations like REGISTER AND LOGIN, PREDICT STUDENT'S PROFILE DETECTION TYPE, VIEW YOUR PROFILE.

7. SCREEN SHOTS





INTERNATIONAL JOURNAL OF APPLIED SCIENCE ENGINEERING AND MANAGEMENT
ISSN 2454-9940
www.ijasem.org
Vol 18, Issue 4, 2024
https://doi.org/10.5281/zenodo.14065961

8.CONCLUSION

Because they offer quick and impartial evaluations of the code that students write and submit, online judge (OJ) systems have received a lot of attention in the context of programming-related courses. Notwithstanding their obvious benefits, OJ systems often just tell the teacher and student whether the given code completed the assignment satisfactorily. Although this restriction is somewhat acceptable, it would be helpful for these systems to recover more data that might eventually be used to identify student habits, behavioural patterns, or profiles associated with task success (or failure), among other things. Although these kinds of insights are considered important in the realm of education, it should be noted that current OJ-based techniques are unable to handle the process.

Using the discipline of Educational Data Mining (EDM), this work attempts to address this restriction. In order to do this, the proposal takes into account modelling student behaviour using code submissions and learning-based approaches from the EDM field, including Multi-Instance Labelling (MIL) and traditional Machine Learning (ML) formulations. Furthermore, we suggest using Explainable Artificial Intelligence (XAI) to provide interpretable feedback, as these frameworks typically fail to deliver the method's desired result of human-understandable input.

A case study using data collected from a computer science degree course on programming has been used to assess this technique. With over 2,500 submissions from about 90 different students—representing all students taking the commented course and about 80% of the total enrollment—this collection includes the various submissions to an OJ system of two distinct assignments over the course of three academic years. The findings support the proposal: based just on the behavioural pattern deduced from the submitted work, the model can accurately predict the user outcome (passing or failing the assignment) in terms of statistical significance.

INTERNATIONAL JOURNAL OF APPLIED SCIENCE ENGINEERING AND MANAGEMENT
https://doi.org/10.5281/zenodo.14065961
ISSN 2454-9940
www.ijasem.org
Vol 18, Issue 4, 2024

Additionally, by identifying student groups that are more likely to fail, the idea makes it feasible to give feedback to both the teacher and the student.

In the future, the model will be further validated by expanding the case study's data set and taking into account additional courses that similarly use OJ assessment techniques. Additionally, in order to improve the system's prediction accuracy, we will examine the potential for investigating the use of human factor characteristics derived from, for example, personality, self-efficacy, and motivation tests.

REFERENCES

[1] A. Kurnia, A. Lim, and B. Cheang, "Online judge," Computers & Education, vol. 36, no. 4, pp. 299–315, 2001.

[2] S. Wasik, M. Antczak, J. Badura, A. Laskowski, and T. Sternal, "A survey on online judge systems and their applications," ACM Computing Surveys (CSUR), vol. 51, no. 1, pp. 1–34, 2018.

[3] R. Yera and L. Mart´ınez, "A recommendation approach for programming online judges supported by data preprocessing techniques," Applied Intelligence, vol. 47, no. 2, pp. 277–290, 2017.

[4] B. Cheang, A. Kurnia, A. Lim, and W.-C. Oon, "On automated grading of programming assignments in an academic institution," Computers & Education, vol. 41, no. 2, pp. 121–131, 2003.

[5] L. M. Regueras, E. Verdu, M. F. Munoz, M. A. Perez, J. P. de Castro, and M. J. Verdu, "Effects of competitive e-learning tools on higher education students: A case study," IEEE Transactions on Education, vol. 52, no. 2, pp. 279–285, 2009.

[6] A. Mani, D. Venkataramani, J. Petit Silvestre, and S. Roura Ferret, "Better feedback for educational online judges," in Proceedings of the 6th International Conference on Computer Supported Education, Volume 2: Barcelona, Spain, 1-3 April, 2014. SciTePress, 2014, pp. 176–183.

[7] R. Asif, A. Merceron, S. A. Ali, and N. G. Haider, "Analyzing undergraduate students' performance using educational data mining," Computers & Education, vol. 113, pp. 177–194, 2017.

[8] X. Zhang and Y. Kang, "Examining and predicting teacher professional development by machine learning methods," in International Conference on Neural Computing for Advanced Applications. Springer, 2021, pp. 255–269.

[9] C. C. Gray and D. Perkins, "Utilizing early engagement and machine learning to predict student outcomes," Computers & Education, vol. 131, pp. 22–32, 2019.

[10] J. R. Rico-Juan, A.-J. Gallego, and J. Calvo-Zaragoza, "Automatic detection of inconsistencies between numerical scores and textual feedback in peer-assessment processes with machine learning," Computers & Education, vol. 140, p. 103609, 2019.

[11] S. Alturki, N. Alturki, and H. Stuckenschmidt, "Using educational data

ISSN 2454-9940
www.ijasem.org
Vol 18, Issue 4, 2024
https://doi.org/10.5281/zenodo.14065961

mining to predict students' academic performance for applying early interventions," Journal of Information Technology Education:
Innovations in Practice, vol. 20, no. 1, pp. 121–137, 2021.
[12] J. Foulds and E. Frank, "A review of multi-instance learning assumptions,"
The knowledge engineering review, vol. 25, no. 1, pp. 1–25, 2010.
[13] M.-L. Zhang, "Generalized multi-instance learning: Problems, algorithms
and data sets," in 2009 WRI Global Congress on Intelligent
Systems, vol. 3. IEEE, 2009, pp. 539–543.
[14] S. Anupama Kumar and M. N. Vijayalakshmi, Efficiency of Multiinstance
Learning in Educational Data Mining. Singapore: Springer, 2018, pp. 47–64.
[15] A. Zafra, C. Romero, and S. Ventura, "Multi-instance learning versus singleinstance learning for predicting the student's performance," Handbook of Educational Data Mining, pp. 187–200, 2010.