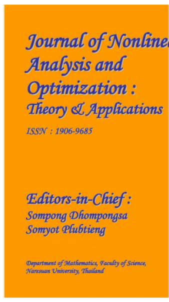


Journal of **N**onlinear **A**nalysis and **O**ptimization
Vol. 15, Issue. 1, No.15 : 2024
ISSN : **1906-9685**

EXPLAINABLE AI-DRIVEN STUDENT PROFILE IDENTIFICATION IN ONLINE JUDGE SYSTEMS

[1]Srujana Kammari,[2]Mr Y.Anitha Reddy
[1]Student,[2]Assistant Professor
Department Of CSE
SVR Engineering College, Nandyal

ABSTRACT

Because Online Judge (OJ) systems provide quick and impartial evaluations of the code students write, they are often taken into consideration in programming-related courses. Based on a rubric, such an assessment often yields a single conclusion, most frequently indicating whether the submission satisfactorily completed the task. However, as this information can be considered inadequate in an educational setting, it would be advantageous for the teacher and the student to have more input about the task's overall growth. By taking into account the additional exploitation of the data collected by the OJ and automatically deriving feedback for the teacher and the student, this study seeks to address this constraint. More specifically, we examine the modeling of student behavior via learning-based methods, including multi-instance learning (MIL) and conventional machine learning formulations. Furthermore, the idea of explainable artificial intelligence (XAI) to provide feedback in a human-readable manner is being considered. The proposal was assessed in light of a case study that included 2500 entries from around 90 different students enrolled in a computer science degree course that dealt with programming. The following outcomes support the proposal: Based only on the behavioral pattern deduced from the submissions made to the OJ, the model can accurately predict the user outcome (passing or failing the assignment). Additionally, the proposal may identify student groupings and profiles that are more likely to fail as well as other pertinent data, which will ultimately provide feedback to the teacher and the student.

I. INTRODUCTION

1.1 About The Project:-

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 OJ systems are successful in the education field because they overcome the main issues associated with the manual evaluation of assignments 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 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 Within this discipline, Machine Learning (ML) is reported as one of the main enabling technologies due to its power and flexibility 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 may not be available. Moreover, other data used to predict student performance, such as socioeconomic background or academic success in other courses may not be usable from an ethical point of view due to the potential biases it would introduce.

1.2 Existing System :-

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.

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.

1.2.1 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.

1.3 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. 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.

1.3.1 ADVANTAGES:-

• Transparency methods, which represent the ones that directly convey the workings of the model.

• 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.

II. LITERATURE SERVEY

Online Judge (OJ) systems are platforms where students solve programming problems and submit their solutions to be evaluated automatically. Understanding student profiles within these systems can provide valuable insights into their learning behaviors, strengths, and areas needing improvement. Explainable

AI (XAI) plays a crucial role in making the decision-making processes of AI models transparent, thereby enhancing the interpretability and trustworthiness of the profiles generated.

Key Themes and Concepts:-

1.Online Judge Systems

• Definition and Functionality*: OJ systems like LeetCode, Codeforces, and HackerRank offer a variety of programming problems and automatically evaluate student submissions for correctness and efficiency.

• Importance in Education: These systems are integral in computer science education, offering real-time feedback and a platform for practice and assessment.

2. Student Profiles in OJ Systems

• Profiling Criteria: Student profiles can be based on various factors such as problem-solving skills, code efficiency, submission patterns, and learning progression.Benefits of Profiling: Identifying different student profiles helps in tailoring educational resources, personalized learning experiences, and targeted interventions.

3. Explainable AI (XAI)

• Definition: XAI refers to methods and techniques that make the output of AI models understandable to humans.

• Relevance to Education: In the context of student profiling, XAI ensures that the profiling process is transparent, thereby helping educators understand the basis of the profiles and trust the recommendations made by AI models.

Literature Review

1. Machine Learning in Education

Applications in OJ Systems: Studies have shown that machine learning models can effectively classify student behaviors and predict performance in OJ systems (e.g., Piech et al., 2015).

• Feature Engineering: Important features include submission time, code correctness, problem difficulty, and edit distance between consecutive submissions.

2. Explainable AI Techniques

• Model-Agnostic Methods: Techniques such as LIME (Local Interpretable Model-agnostic Explanations) and SHAP (SHapley Additive exPlanations) provide insights into model predictions by approximating simpler models or attributing the impact of features (Ribeiro et al., 2016; Lundberg & Lee, 2017).

• Interpretable Models: Decision trees and rule-based systems are inherently interpretable and can be used to create transparent student profiles (Breiman et al., 1984).

3. Profiling Student Behaviors

• Clustering Techniques: Clustering algorithms like k-means and hierarchical clustering have been used to group students based on their problem-solving patterns and learning trajectories (Hung et al., 2010).

• Predictive Models: Models such as logistic regression and neural networks have been employed to predict student success and identify at-risk students, with XAI techniques providing explanations for these predictions (Morsy & Karypis, 2017).

4. Case Studies and Applications

• Personalized Feedback: Research by Rivers and Koedinger (2017) demonstrated the use of XAI to provide personalized feedback to students based on their performance in OJ systems.

• Adaptive Learning Systems: Studies have shown the integration of XAI in adaptive learning systems, where student profiles are used to dynamically adjust the difficulty and type of problems presented (Desmarais & Baker, 2012).

The integration of XAI into the identification of student profiles within OJ systems presents a promising avenue for enhancing educational outcomes. By making AI-driven insights transparent and interpretable, educators can better understand and trust the profiling process, leading to more effective and personalized educational interventions. Future research should focus on developing more sophisticated XAI techniques tailored to educational contexts and exploring their impact on student learning experiences and outcomes.

III. SYSTEM DESIGN ARCHITECTURE

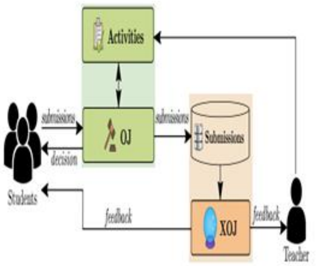


Fig: Graphical representation of the scheme proposed

IV.ALGORITHMS:-

These Algorithms are used in this project:

DECISION TREE CLASSIFIERS:-

Decision tree classifiers are used successfully in many diverse areas. Their most important feature is the capability of capturing descriptive decision making knowledge from the supplied data. Decision tree can be generated from training sets. The procedure for such generation based on the set of objects (S), each belonging to one of the classes C1, C2, …, Ck is as follows:

Step 1. If all the objects in S belong to the same class, for example Ci, the decision tree for S consists of a leaf labeled with this class

Step 2. Otherwise, let T be some test with possible outcomes O1, O2,…, On. Each object in S has one outcome for T so the test partitions S into subsets S1, S2,… Sn where each object in Si has outcome Oi for T. T becomes the root of the decision tree and for each outcome Oi we build a subsidiary decision tree by invoking the same procedure recursively on the set Si.

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.

K-NEAREST NEIGHBORS (KNN):-

- Simple, but a very powerful classification algorithm
- Classifies based on a similarity measure
- Non-parametric
- Lazy learning
- Does not “learn” until the test example is given
- Whenever we have a new data to classify, we find its K-nearest neighbors from the training data

Example

• Training dataset consists of k-closest examples in feature space

• Feature space means, space with categorization variables (non-metric variables)

• Learning based on instances, and thus also works lazily because instance close to the input vector for test or prediction may take time to occur in the training dataset.

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 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.

NAIVE 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 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, which will translate into several hyperplanes' meeting this requirement.

7.IMPLEMENTATION

7.1 MODULES AND DESCRIPTION

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.

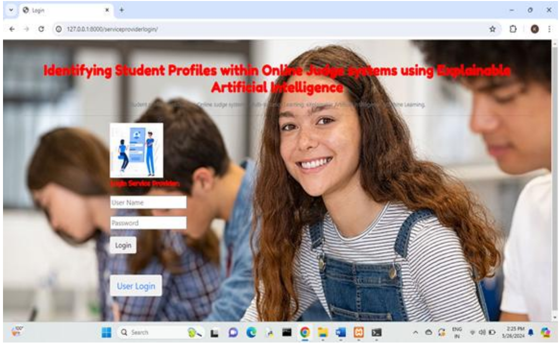
Remote User:-

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.

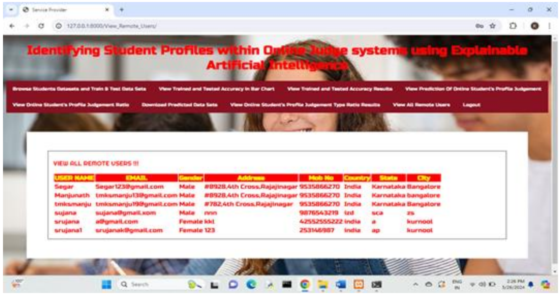
V. SCREENSHOTS

Home page:-

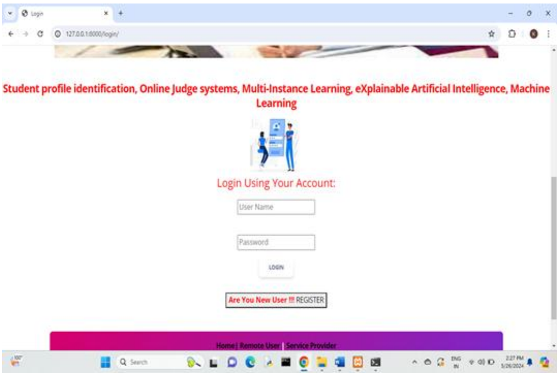
Login:-



Service provider:-

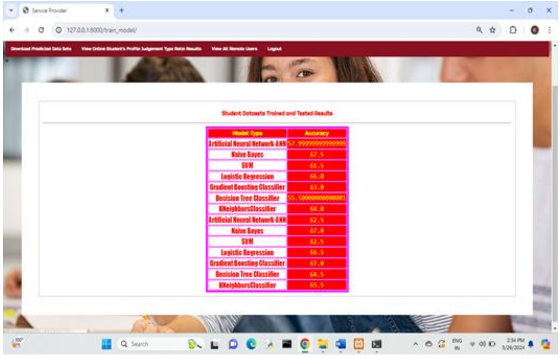


Remote user registration:-



Registration details remote user:-

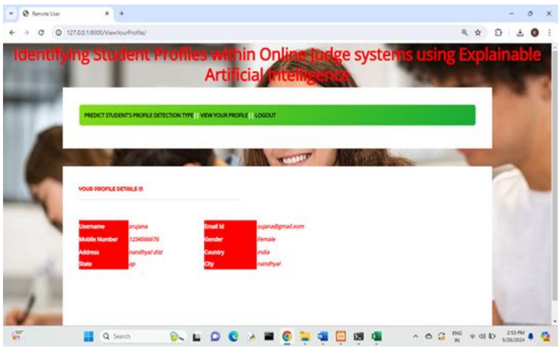
Student Datasets Trained And Tested Results :-



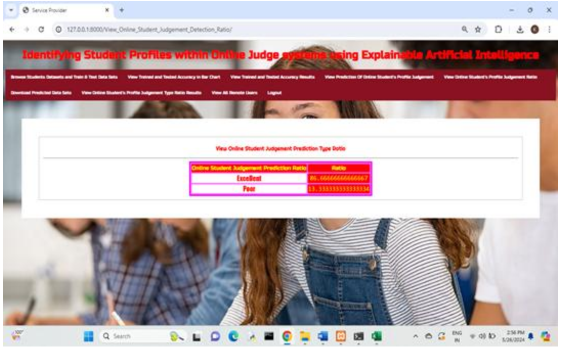
Prediction of online students profile judgement detection:-

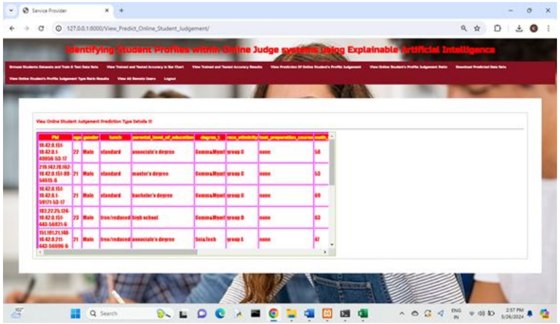
Prediction results:-

Remote User Details:-

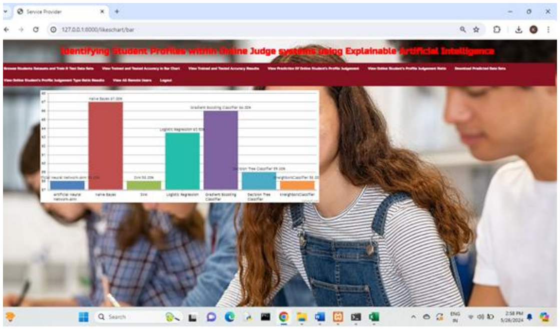


View online student judgement ratio:-

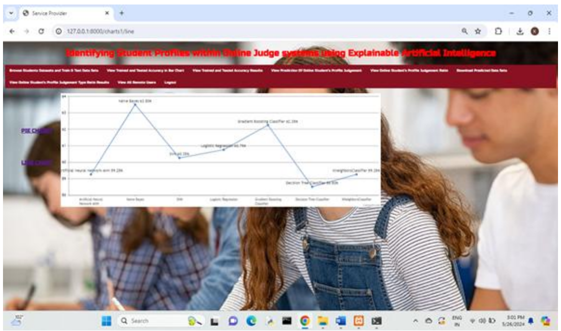




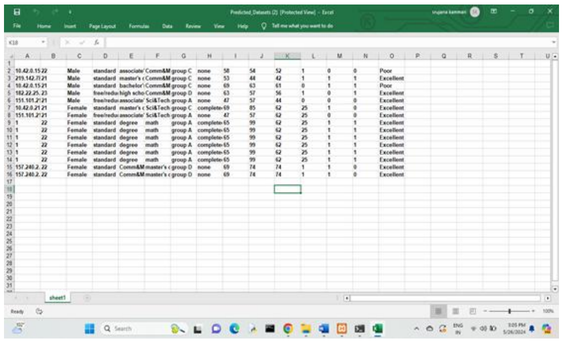
View trained and tested accuracy in barchart:-



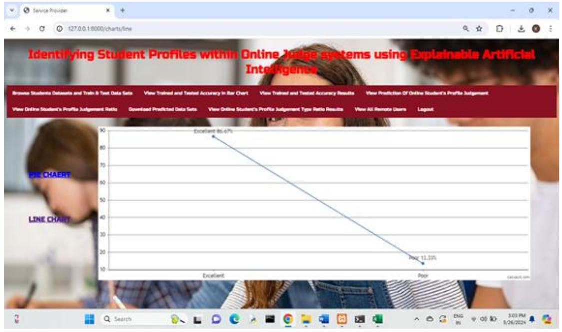
View trained and tested results :-



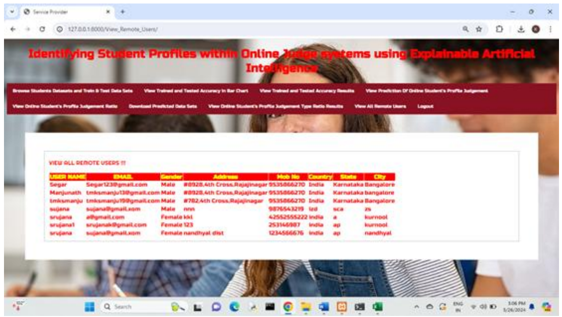
Download predicted dataset:-



View online profile judgement type ratio results:-



9.16 All remote users:-



VI. CONCLUSION

Online Judge (OJ) systems have been largely considered in the context of programming-related courses as they provide fast and objective assessments of the code developed and submitted by the students.

Despite their clear advantages, OJ systems do not generally provide the student nor the instructor with any feedback from the actual submission besides whether the provided code successfully accomplished the assignment.

Future work considers the further validation of the model, both increasing the amount of data of the case of study as well as considering other alternative courses that also resort to OJ evaluation methods. In addition, we will consider the possibility of exploring the use of human factor characteristics drawn from, for instance, personality, self-efficacy, and motivation tests to boost the prediction accuracy of the system.

11. FUTURE ENHANCEMENT

Future enhancement for this project by integrating explainalble AI, the system can not only evaluate students coding abilities but also provide transparent insights into their problem solving approaches,learning patterens,and areas needing improvement .

This Transparency fosters trust and allows educators to tailor feedback and resources more effectively.additionally,explainable AI can identify diverse learning styles and predict potential challenges,enabling proactive spport.such enhancements ensure a more inclusive, adaptive learning environment that empowers students through clear,actionable feedback and guidance.

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 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.
16. T. Kom´arek, J. Brabec, and P. Somol, "Explainable multiple instance learning with instance selection randomized trees," in Joint European Conference on Machine Learning and Knowledge Discovery in Databases. Springer, 2021, pp. 715–730.
17. N. Burkart and M. F. Huber, "A survey on the explainability of supervised machine learning," Journal of Artificial Intelligence Research, vol. 70, pp. 245–317, 2021.
18. A. B. Arrieta, N. D´ıaz-Rodr´ıguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. Garc´ıa, S. Gil-L´opez, D. Molina, R. Benjamins et al., "Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI," Information Fusion, vol. 58, pp. 82–115, 2020.
19. J. B. Hext and J. W. Winings, "An automatic grading scheme for simple programming exercises," Commun. ACM, vol. 12, no. 5, p. 272–275, May 1969.
20. R. C. Carrasco, J. R. Rico-Juan, and M. A´ . Varo´, "Aprendizaje de algoritmia mediante desaf´ıos de programaci´on," in XVI Jornadas de Ense˜nanza Universitaria de la Inform´atica. Universidade de Santiagode Compostela. Escola T´ecnica Superior d'Enxe˜nar´ıa, 2010, pp. 519–522.
21. J. L. Bez, N. A. Tonin, and P. R. Rodegheri, "Uri online judge academic: A tool for algorithms and programming classes," in 2014 9th International Conference on Computer Science Education, 2014, pp. 149–152.
22. L. Wen-xin and G. Wei, "Peking university oneline judge and its applications," Journal of Changchun Post and Telecommunication Institute, vol. 2, 2005.
23. C. A. Higgins, G. Gray, P. Symeonidis, and A. Tsintsifas, "Automated assessment and experiences of teaching programming," J. Educ. Resour. Comput., vol. 5, no. 3, p. 5–es, Sep. 2005.
24. H. Sun, B. Li, and M. Jiao, "Yoj: An online judge system designed for programming courses," in 2014 9th International Conference on Computer Science Education, 2014, pp. 812–816.
25. A. Kosowski, M. Małafiejski, and T. Noi´nski, "Application of an online judge & contester system in academic tuition," in Advances in Web Based Learning – ICWL 2007, H. Leung, F. Li, R. Lau, and Q. Li, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 343–354.