IJMIC Vol. 3, Issue 1, 28 - 45

Research Article

AutoML Algorithms for Predicting Students' Dropout and Academic Success in Higher Education

Siti Hajar Datu Ali Nafiah[1], Jason Teo[2,3*]

[1]Faculty of Computing and Informatics, Universiti Malaysia Sabah, Sabah, Malaysia

[2]Creative Advance Machine Intelligence Research Centre, Faculty of Computing and Informatics, Universiti Malaysia Sabah, Kota Kinabalu, Malaysia

[3]Evolutionary Computing Laboratory, Faculty of Computing and Informatics, Universiti Malaysia Sabah, Kota Kinabalu, Malaysia

*Corresponding author: Jason Teo, jtwteo@ums.edu.my

Received: [Jan 3, 2026]
Accepted: [June 19, 2026]
Published: [July 30, 2026]

Abstract – Challenges with high dropout rates and low academic achievement continue to be a major issue for higher education institutions. To deal with these challenges, it is essential to identify the underlying causes and develop strategic solutions to support student success. Machine learning techniques are powerful tools for predicting student academic performance and enabling timely, targeted interventions. In this study, the goal is to test Automated Machine Learning (AutoML) frameworks that can predict student dropout and academic success. The dataset included students' demographic, socio-economic, and academic information gathered from institutional records. H2O AutoML, TPOT, PyCaret, and AutoGluon were evaluated, as these tools automate important parts of the machine learning pipeline, making it easier to train and select models. Evaluation metrics including accuracy, precision, recall, and F1-score, as well as training time and CPU usage, were used to assess model performance and computational efficiency. Three main experiments were conducted: the first evaluated model diversity, the second observed the effect of cross-validation fold size, and the third assessed different search space configurations. In the first experiment, TPOT obtained the highest accuracy of 90.04%. In the second experiment, using 10-fold cross-validation, H2O AutoML obtained the best accuracy of 91.38%. In the third experiment, TPOT maintained its accuracy at 90.04% across all configurations. A comparison between the conventional ExtraTreesClassifier and TPOT's AutoML ExtraTreesClassifier showed comparable performance, with the conventional model achieving slightly higher accuracy of 90.20% and recall of 94.44%, while TPOT demonstrated superior precision of 90.24%. These results demonstrate that AutoML frameworks are effective tools for early identification of at-risk students, offering competitive performance with significantly reduced manual effort. Future studies could explore deploying these models within academic advising systems.

Keywords: AutoML Framework, Machine Learning, Conventional Framework, Student Dropout, Higher Education

1. INTRODUCTION

Higher education institutions continue to face the challenges of student dropout and academic failure, which affect not only individual students but also the institutions themselves. The increasing number of student dropouts imposes significant costs on the economy and productivity. According to Realinho et al. [1], among the 4,424 students enrolled at the Polytechnic Institute of Portalegre between 2008 and 2019, approximately 32% dropped out of their programmes, representing a substantial proportion of learners who did not complete their studies. Therefore, creating a predictive model to identify at-risk students early has become essential for effective intervention. By examining a variety of data sources, including demographic, socio-economic, and academic performance factors, machine learning (ML) offers a promising method of predicting student dropouts [2], [3], [4]. Based on previous studies, researchers have used various methods, including decision trees, support vector machines, and ensemble learning, to develop more accurate dropout prediction models [5], [6], [7].

The development of Automated Machine Learning (AutoML) frameworks has simplified the construction of predictive models by automating feature selection, model selection, and hyperparameter tuning. Several AutoML frameworks, such as H2O AutoML, TPOT, PyCaret, and AutoGluon have been applied in numerous studies, demonstrating their usefulness across many domains, including disease diagnosis [8], stock prices prediction [9], and quality assurance of water [10]. Using AutoML provides a strong foundation for predicting student dropout and academic success [11], [12].

H2O AutoML is known for its ensemble learning method which uses multiple base models to achieve robust results [10]. AutoSklearn is an AutoML framework that has been used in numerous studies, such as to estimate the compressive strength of concrete [13]. Researchers have looked into using AutoKeras in deep learning, especially for diagnosing medical images, where it has demonstrated high accuracy in classifying diseases [8]. TPOT is based on genetic programming that has been used to make ML pipelines better at predictive analysis [14]. In addition to AutoML, numerous studies have examined conventional machine learning to predict student dropout and academic success such as XGBoost, which has yielded strong results [15]. Some studies have used ensemble learning methods such as Random Forest and Gradient Boosting [16], [17]. In some cases, logistic regression has also been found to be a good starting point for predicting student dropouts [16].

In this study, AutoML frameworks, namely H2O AutoML, TPOT, AutoGluon, and PyCaret, are evaluated for predicting student dropout and academic success through experiments examining model diversity, cross-validation strategy, and search space configuration.

AutoML automates feature engineering, hyperparameter tuning, and model selection, thereby reducing the manual effort required compared to conventional machine learning methods. Additionally, the computational time and resource consumption of each framework are measured to assess scalability with the educational datasets used. Beyond benchmarking framework performance, this study contributes a practical decision-making framework for selecting AutoML tools in educational contexts by linking each framework's predictive performance, resource requirements, and interpretability characteristics to specific deployment scenarios, such as resource-constrained institutions or settings requiring rapid prototyping. Furthermore, by examining model generalizability through cross-validation and discussing the operationalization of the proposed models in a different academic context, this study extends beyond a single-dataset comparison to offer guidance on the transferability and practical adoption of AutoML-based dropout prediction systems in higher education. These contributions help fill the gap in the literature regarding how AutoML frameworks can be evaluated and selected for real-world educational applications, supporting the development of early intervention strategies in higher education.

2. METHODOLOGY

Software and Hardware Specification

This study uses Windows 10 Home Single Language as the operating system, Google Colab as the coding environment, and Python 3.11.11 for developing and executing machine learning models. The hardware requirements include an Intel Core i3-10110U CPU @ 2.10GHz-2.59 GHz, and 8.0 GB of RAM.

Data Acquisition

The dataset utilized in this project was sourced from the UCI repository, a collection of databases in various domains used by the machine learning community for the empirical analysis of machine learning algorithms. Mónica Vieira Martins is credited as one of the publishers of the dataset, which is accessible at https://archive.ics.uci.edu/dataset/697/predict+students+dropout+and+academic+success.

The dataset is accompanied by an introductory paper describing the techniques and machine learning approaches used for prediction. The dataset which is structured in the format of a comma separated value (CSV) file comprises 35 features of 4424 students enrolled in undergraduate courses of Polytechnic Institute of Portalegre, Portugal between academic years 2008/09–2018/2019 and from different undergraduate degrees, such as agronomy, design, education, nursing, journalism, management, social service, and technologies. With 34 input features covering diverse aspects of student profiles and environmental influences, the dataset provides a rich, multidimensional source of information for predictive modelling, along with one target feature representing student dropout and academic success. The independent variable in this paper has 3 main factors which are the demographic, socio-economic and academic factor whereas the dependent variable is “Target”.

Data Pre-Processing

Data preparation is a critical first step to ensure the data is clean and ready for the modelling phase. In this study AutoML will be leveraged as a proposed method. Although AutoML tools such as H2O AutoML can automate many aspects of data preparation, manual preprocessing was applied to ensure a fair and comparable evaluation between AutoML and conventional frameworks. This included thorough data cleaning and error correction, domain-specific feature engineering, fine-tuning optimization, and developing a deeper understanding of the dataset.

To create a robust evaluation framework, the dataset will be divided into three subsets which are training, testing and validation. Each of the subsets has 70%, 15%, and 15% respectively of the main dataset. The training set is used to train the models, the validation set is used for hyperparameter tuning and to prevent overfitting, and the testing set is used to evaluate final model performance. This approach ensures that models can generalize well to the unseen data and provide reliable information for comparing different AutoML against the conventional framework.

AutoML Framework

AutoML frameworks are designed to automate the process of applying machine learning models to real-world problems. These frameworks handle tasks such as feature engineering, hyperparameter optimization, and model selection without requiring extensive manual intervention [18]. By eliminating much of the trial-and-error traditionally involved in machine learning model development, AutoML frameworks enable more efficient and scalable predictive analytics.

H2O AutoML employs a stacked ensemble approach, combining models like gradient boosting machines, deep neural networks, and random forests in a structured pipeline that prioritizes predictive accuracy [19]. Its standout feature lies in its ability to produce highly optimized ensemble models with minimal user intervention, while also offering interpretability tools that enhance understanding of feature impact, which is an essential aspect in applications requiring model transparency. Meanwhile, TPOT leverages genetic programming to evolve entire machine learning pipelines, automatically navigating through various models and feature engineering steps to identify optimal configurations [20]. TPOT constructs a pipeline that includes preprocessing and feature selection which makes it versatile. AutoGluon employs an automated tuning process that accommodates diverse datasets by incorporating meta-learning and multi-layer stacking to enhance predictive performance [21]. Its ability to handle data with minimal preprocessing, along with native support for text, image, and tabular formats, makes it practical for users with minimal knowledge in machine learning. Lastly, PyCaret simplifies machine learning workflow through its low-code interface, by automating tasks like data preprocessing, modelling, and evaluation to enable faster experimentation and deployment where ease of use and speed are priorities [22].

Each AutoML framework presents distinct trade-offs in terms of execution speed, interpretability, and scalability. The choice of framework depends on factors such as dataset complexity and available computational resources. The goal of this study is to evaluate whether these frameworks can effectively predict student dropout and academic success.

Experimental Techniques

The experimental technique used in this study was designed to compare the performance of different AutoML frameworks for predicting student dropout and academic success as shown in Figure 1. The experiments were structured to answer key research questions regarding model diversity, cross-validation impact, and search space size.

The first experiment investigated model diversity across AutoML frameworks, evaluating how the variety of generated models affects predictive performance. The next experiment investigated the impact of cross-validation fold size on model performance. Cross-validation is a data resampling method to assess the generalization ability of a predictive model and to prevent overfitting [23]. It is a crucial component to have in AutoML frameworks as it helps to estimate the model generalization ability. In this study different cross-validation folds were tested which are the 3-fold, 5-fold and 10-fold to determine how varying validation strategies influence the predictive accuracy of AutoML models. However, it is important to note that higher fold counts require greater computational resources, which should be considered based on available resource allocation. Lastly, the third experiment examined the effect of search space size on model performance. This experiment tested multiple search space configurations that range from small to bigger search spaces. The objective was to determine the trade-off between search space size, model accuracy and the computational efficiency.

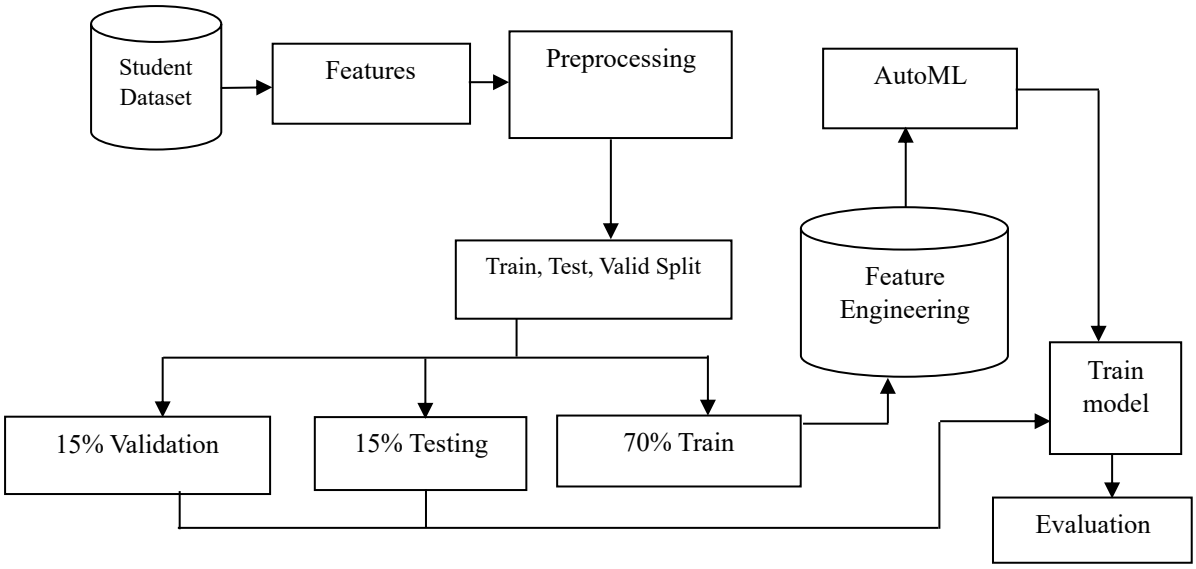

Figure 1: Methodology of the Student Dropout and Academic Success Prediction.

Conventional Framework

After conducting extensive experiments across different AutoML frameworks, the best model was identified and will be used as a baseline model to define the conventional framework. A conventional framework typically involves manual tuning, feature engineering, and optimization, all of which require domain expertise. It is unlike AutoML which automates these steps. By leveraging insights gained from AutoML experiments, the model selection criteria can be refined which will improve both performance and efficiency.

Performance Metric

The study evaluates model performance using standard classification metrics such as accuracy, precision, recall, and F1-score. Accuracy measures the overall correctness of predictions, while precision and recall provide insights into false positives and false negatives. The F1-score balances these two metrics, making it a suitable measure for datasets with imbalanced class distributions.

$$Accuracy = \frac{TP+TN}{TP+FP+FN+TN} \quad (1)$$

$$Precision = \frac{TP}{TP+FP} \quad (2)$$

$$Recall = \frac{TP}{TP+FN} \quad (3)$$

$$F1-Score = \frac{2\times precision\times recall}{precision+recall} \quad (4)$$

Where,

True positive (TP) defined actual value and predicted value are both positive.

True negative (TN) defined actual value and predicted value are both negative.

False positive (FP) defined actual value is negative but predicted value is positive.

False negative (FN) defined actual value is positive but predicted value is negative.

AutoML frameworks often involve complex algorithms and computationally intensive tasks, therefore an additional evaluation metric used in this project to compare different AutoML frameworks is the time metric and resources usage. The time metric is used to measure the computational efficiency and scalability of different AutoML frameworks and algorithms. It assesses the time taken by each framework to complete the AutoML process, including data preprocessing, feature selection, model training, and evaluation. Training time is a crucial consideration, as it directly impacts the feasibility of deploying AutoML solutions in real-world educational settings.

3. RESULTS & DISCUSSIONS

3.1 Experiment 1: Model diversity

In this experiment, two configurations were used. Configuration 1 used all models available in the framework library. Configuration 2 was restricted to only three models: Logistic Regression (LR), XGBoost, and Random Forest (RF).

As shown in **Table 1**, H2O AutoML gave the best performance with accuracy of 89.67%, precision of 90.57% as well as 89.57% of F1-Score indicating its high effectiveness across a diverse set of models. PyCaret has the highest recall of 92.28% but had the lowest precision which is 86.64% compared to other frameworks. TPOT and AutoGluon had the same accuracy of 88.98%. However, TPOT achieved a slightly better precision of 89.41%, while AutoGluon recorded a slightly better recall of 88.99%.

Table 1: Configuration 1: Uses all models available within the respective framework libraries.

AutoML	Accuracy	Precision	Recall	F1-Score
H2O AutoML	**89.67**	**90.57**	89.47	**89.57**
TPOT	88.98	89.41	88.83	88.92
AutoGluon	88.98	89.17	88.99	88.97
PyCaret	89.18	86.64	**92.28**	89.47

In Configuration 2, TPOT delivered the best performance as shown in **Table 2** with the highest accuracy, precision and F1-Score of 90.04%, 90.24% and 90.07% respectively. PyCaret had the highest recall of 92.28% but lower precision of 86.64%. H2O AutoML showed a decline in performance when limited to fewer models, with accuracy of 88.30%.

Table 2: Configuration 2: Limited the frameworks to three specific models: Logistic Regression (LR), XGBoost, and Random Forest (RF).

AutoML	Accuracy	Precision	Recall	F1-Score
H2O AutoML	88.30	88.70	88.16	88.23
TPOT	**90.04**	**90.24**	89.94	**90.07**
AutoGluon	87.78	87.96	87.78	87.77
PyCaret	89.18	86.64	**92.28**	89.47

3.2 Experiment 2: Cross-Validation Fold Size

This experiment was conducted to observe the effect of varying the number of cross-validation folds, for example 3-fold, 5-fold, 10-fold on the model performance of each AutoML framework. Cross-validation is a technique used to improve model generalizability by training and validating the model on different data splits.

In the first fold setting which is 3-folds, as shown in **Table 3**, H2O AutoML delivered the highest mean accuracy and F1-Score of 90.14% ± 1.34% and 90.66% ± 1.11%, indicating that it can achieve robust performance even with fewer cross-validation folds. Meanwhile, TPOT achieved a solid mean accuracy of 88.62% ± 0.44% and F1-Score of 88.99% ± 0.42%, favoring recall at 92.58% ± 0.90%. AutoGluon, on the other hand, had slightly lower performance with accuracy of 87.68% ± 1.13% and F1-Score of 88.23% ± 1.02%. However, it performed comparably in terms of recall at 92.45% ± 0.80%, even though precision was only 84.39% ± 1.38%. PyCaret achieved a similar accuracy of 87.69% ± 1.15% and F1-Score of 88.57% ± 1.25%, but stood out with the highest recall across all frameworks at 95.25% ± 2.71%, though at the cost of the lowest precision of 82.77% ± 0.49%, also showing the highest variability in recall across folds.

Table 3: Hyperparameter: Cross-Validation 3

AutoML	Accuracy	Precision	Recall	F1-Score
H2O AutoML	**90.14 ± 1.34**	**86.62 ± 1.85**	95.10 ± 0.35	**90.66 ± 1.11**
TPOT	88.62 ± 0.44	85.67 ± 0.72	92.58 ± 0.90	88.99 ± 0.42
AutoGluon	87.68 ± 1.13	84.39 ± 1.38	92.45 ± 0.80	88.23 ± 1.02
PyCaret	87.69 ± 1.15	82.77 ± 0.49	**95.25 ± 2.71**	88.57 ± 1.25

Next, increasing the fold to 5 as shown in **Table 4**, TPOT improved its accuracy to 88.91% ± 1.58% and F1-Score to 89.20% ± 1.45%, with a slightly better precision of 86.51% ± 2.53%, indicating moderate stability with increasing cross-validation folds. AutoGluon's performance remained consistent with accuracy of 88.08% ± 0.38% and F1-Score of 88.54% ± 0.38%, notably exhibiting the smallest standard deviation across all frameworks, suggesting the most stable performance across folds. H2O AutoML maintained strong performance with 90.71% ± 1.04% accuracy and 91.03% ± 1.06% F1-Score, showcasing its effectiveness across increasing fold configurations. PyCaret delivered accuracy of 87.90% ± 2.38% and F1-Score of 89.34% ± 2.03%, again achieving the highest recall of 95.66% ± 3.45% but with the greatest variability in both precision (83.03% ± 3.73%) and recall, reflecting inconsistency across folds.

Table 4: Hyperparameter: Cross-validation 5.

AutoML	Accuracy	Precision	Recall	F1-Score
H2O AutoML	**90.71 ± 1.04**	**88.33 ± 1.38**	93.94 ± 2.01	**91.03 ± 1.06**
TPOT	88.91 ± 1.58	86.51 ± 2.53	92.12 ± 2.17	89.20 ± 1.45
AutoGluon	88.08 ± 0.38	85.17 ± 0.79	92.21 ± 1.08	88.54 ± 0.38
PyCaret	87.90 ± 2.38	83.03 ± 3.73	**95.66 ± 3.45**	89.34 ± 2.03

Lastly, by increasing the fold to 10 as shown in **Table 5**, TPOT further improved its mean accuracy to 89.07% ± 2.15% and F1-Score to 89.31% ± 2.04%, also demonstrating slightly greater precision of 86.94% ± 2.85% and recall of 91.92% ± 2.94% with higher fold counts. AutoGluon also showed slight

improvement in performance with 88.45% ± 0.52% accuracy and 88.89% ± 0.47% F1-Score, alongside marginal increases in precision of 85.48% ± 0.72% and recall of 92.60% ± 0.43%, continuing to exhibit the lowest standard deviation across all metrics, confirming its consistency. Across all frameworks, H2O AutoML again achieved the best performance with 91.38% ± 1.27% accuracy and 91.68% ± 1.12% F1-Score, confirming its superiority with larger fold sizes. PyCaret, while achieving the highest recall of 95.69% ± 4.10%, showed the greatest variability across all metrics, particularly in precision (83.97% ± 4.16%) and accuracy (88.54% ± 2.93%), suggesting sensitivity to fold composition under 10-fold cross-validation.

Table 5: Hyperparameter: Cross-validation 10.

AutoML	Accuracy	Precision	Recall	F1-Score
H2O AutoML	**91.38 ± 1.27**	**89.02 ± 1.50**	94.55 ± 2.01	**91.68 ± 1.12**
TPOT	89.07 ± 2.15	86.94 ± 2.85	91.92 ± 2.94	89.31 ± 2.04
AutoGluon	88.45 ± 0.52	85.48 ± 0.72	92.60 ± 0.43	88.89 ±0.47
PyCaret	88.54 ± 2.93	83.97 ± 4.16	**95.69 ± 4.10**	89.35 ± 2.70

Although the accuracy differences between frameworks appear numerically small in some configurations, it is important to determine whether these differences are statistically meaningful. A Friedman test was conducted to compare the AutoML frameworks, measured under the same conditions, namely the 10 cross-validation folds. The Friedman test is a non-parametric statistical test designed for comparing three or more related groups, in this case, the four AutoML frameworks, evaluated under matched conditions, namely the same 10 cross-validation folds. Rather than comparing raw accuracy values directly, the test ranks the frameworks within each fold and evaluates whether these rankings are consistent across folds. The test produces a chi-squared (χ^2) statistic, which reflects the magnitude of rank-sum differences between frameworks across the 10 folds. A larger χ^2 indicates that the frameworks differ substantially in their ranked performance, making it less likely the result occurred by chance. A smaller χ^2 suggests the rank-sum differences are small and consistent with random variation. The accompanying p-value is the probability of observing rank-sum differences at least this large if all four frameworks performed equivalently — a p-value below 0.05 leads to rejection of that null hypothesis. Applying this test to the per-fold accuracy scores of H2O AutoML, TPOT, AutoGluon, and PyCaret under 10-fold cross-validation yielded $\boldsymbol{X^2}$**= 11.64** and **p = 0.0087.** Since the p-value is below the threshold of 0.05, the null hypothesis that all four frameworks perform equally is rejected, indicating that the AutoML framework has a statistically significant effect on the predictive accuracy, and that the performance differences reported in **Table 5** are unlikely to be due to random variation across folds.

3.3 Search Space Size

The third experiment aimed to understand how changing the search space size (small, medium, large) affects the performance of the AutoML frameworks. A larger search space typically includes a greater number of models and combinations, which can lead to better performance but requires more computation time and resources.

3.3.1 All model available for the AutoML framework

In this experiment, using all models available within the AutoML framework, as shown in **Table 6** Configuration 1, H2O AutoML achieved the highest accuracy of 89.67% and precision of 90.57% showcasing the ability to select optimal models within smaller search spaces. The recall and F1-Score were also high, which is around 89% indicating a well-performing framework. However, in

Configuration 2, H2O AutoML experienced a decline in performance with accuracy dropping to 87.56% and F1-Score to 87.55%. This may be due to the resource limitation on the current device which caused the framework to stop loading all models due to the extensive computational demand. Meanwhile TPOT showed competitive results across all configurations. In Configuration 1, it achieved an accuracy of 88.98% and balanced F1-Score of 88.92% with precision and recall values of 89.41% and 88.83% respectively. These numbers were slightly lower than H2O AutoML, but TPOT demonstrated its ability to handle a variety of search space configurations. Moving on to Configuration 2 in **Table 7**, TPOT showed improvement with an accuracy of 89.14% and F1-Score of 89.07%, indicating its effective exploration of the expanded search space. In Configuration 3 as shown in **Table 8**, TPOT maintained consistent performance compared to the previous configuration.

Table 6: Configuration 1: 50 model.

AutoML	Accuracy	Precision	Recall	F1-Score
H2O AutoML	**89.67**	**90.57**	89.47	89.57
TPOT	88.98	89.41	88.83	88.92
AutoGluon	88.38	88.72	88.39	88.36
PyCaret	89.62	87.26	**92.95**	**89.98**

Table 7: Configuration 2: 100 model.

AutoML	Accuracy	Precision	Recall	F1-Score
H2O AutoML	87.56	87.57	87.55	87.55
TPOT	89.14	**89.58**	88.97	89.07
AutoGluon	88.83	89.15	88.84	88.81
PyCaret	**89.26**	86.41	**93.21**	**89.67**

Table 8: Configuration 3: 150 model.

AutoML	Accuracy	Precision	Recall	F1-Score
H2O AutoML	88.32	88.38	88.30	88.31
TPOT	89.14	**89.53**	88.98	89.07
AutoGluon	88.53	88.81	88.54	88.51
PyCaret	**89.91**	87.34	**93.46**	**90.25**

3.3.2 Specific model (Random Forest, XGBoost, Logistic Regression)

Among the individual models examined, TPOT consistently outperformed the others. In Configuration 1 as shown in **Table 9**, it achieved an accuracy of 90.04% and F1-Score of 90.01%, which indicates the strong adaptability to specific model constraints. In following configurations as shown in **Table 10** and **Table 11**, TPOT continued to lead, maintaining its position with accurate and reliable results. H2O AutoML on the other hand, while it is one of the strong frameworks, in this experiment it demonstrated stable but lower performance compared to TPOT, consistently achieving accuracy and F1-Score around

88%. PyCaret excels in recall, and AutoGluon manages to maintain steady but relatively lower performance across all the configurations.

Table 9: Configuration 1: 50 model.

AutoML	Accuracy	Precision	Recall	F1-Score
H2O AutoML	88.30	88.70	88.16	88.23
TPOT	**90.04**	**90.24**	89.94	**90.01**
AutoGluon	88.38	88.64	88.39	88.36
PyCaret	89.52	87.50	**92.24**	89.78

Table 10: Configuration 2: 100 model.

AutoML	Accuracy	Precision	Recall	F1-Score
H2O AutoML	88.60	89.51	88.40	88.49
TPOT	**90.04**	**90.24**	89.94	**90.01**
AutoGluon	88.53	88.70	88.54	88.52
PyCaret	89.30	87.09	**92.30**	89.60

Table 11: Configuration 3: 150 model.

AutoML	Accuracy	Precision	Recall	F1-Score
H2O AutoML	88.60	89.51	88.40	88.49
TPOT	**90.04**	**90.24**	89.94	**90.01**
AutoGluon	88.08	88.41	88.09	88.06
PyCaret	89.36	87.89	**91.33**	89.56

Overall, the finding highlights the strength and limitations of each AutoML framework in different configurations of model setting and search space, with TPOT shown as the top performer, closely followed by H2O AutoML which performed really well in certain configurations. Meanwhile PyCaret showed its ability to prioritize recall and AutoGluon faced challenges with smaller search spaces.

3.4 Conventional Framework vs AutoML Framework

Both Conventional and AutoML framework tools are essential for building a predictive analysis model. The AutoML framework has changed how data scientists do machine learning tasks by automating steps for instance model selection, optimization, and feature engineering. However, conventional frameworks offer more manual, in-depth control over model customization and fine-tuning. To evaluate the strengths and weaknesses of both methodologies, this section contrasted the performance of a traditional ExtraTreesClassifier with that of an AutoML ExtraTreesClassifier utilizing TPOT.

Performance Comparison

The comparison of the conventional ExtraTreesClassifier and TPOT ExtraTreesClassifier is based on several evaluation metrics: accuracy, precision, recall, and F1-score. These metrics provide insights into the models' ability to correctly predict instances, minimize false positives, and capture true positives.

Based on **Table 12**, the conventional ExtraTreesClassifier performed slightly better in terms of overall accuracy, reaching 90.20%, suggesting that the conventional model may deliver marginally stronger predictions overall. However, TPOT recorded a higher precision of 90.24% compared to 87.53% for the conventional model, indicating that TPOT is more reliable in cases where precision matters most, as it tends to produce fewer false positives. On the other hand, the conventional model achieved a higher recall of 94.44%, making it more effective when the goal is to capture as many true positives as possible. In terms of F1-score, the conventional model again held a slight edge at 90.86%, with TPOT close behind at 90.01%. Although the conventional model maintains a small advantage across most metrics, both approaches demonstrate robust and comparable overall performance.

Table 12: Conventional vs AutoML Evaluation Metric Result

Frameworks	Accuracy	Precision	Recall	F1-Score
Conventional ExtraTreesClassifier	**90.20**	87.53	**94.44**	**90.86**
TPOT ExtraTreesClassifier	90.04	**90.24**	89.94	90.01

Implications for Use

The contrast approaches of conventional and AutoML framework bring an important implication for its suitability across different study cases. The strength of AutoML lies in its ability to automate the processes, for example model selection, hyperparameter tuning, effectively reducing the need for manual intervention. This allows the development and production of high-performance models, for instance, TPOT higher precision which reflects its effectiveness in minimizing false positives, a critical factor in reducing misclassification costs. Furthermore, AutoML frameworks are particularly advantageous for users with limited expertise in machine learning, enabling them to build reliable models without deep technical knowledge. The conventional framework is without a doubt a useful method, particularly in scenarios where recall is crucial. The conventional ExtraTreesClassifier, with its higher recall, excels in applications where capturing all true positives is critical. In such cases where cost of false negatives is high, conventional approaches provide more reliable results.

3.5 Resource usage and training time

AutoML frameworks are designed to automate the process of applying machine learning to real-world problems. Each framework has its own strengths, and evaluating their performance regarding resource utilization is essential for choosing the most suitable tool for particular tasks.

Experiment 1: Model Diversity

The execution time of H2O AutoML in Configuration 1 (all models in the framework) was 5483.19 seconds, as shown in **Figure 2**. This is much longer than TPOT (294.52 seconds) and AutoGluon (51.05 seconds). With 155.76 seconds, PyCaret also did fairly well. However, execution times for all

frameworks decreased in Configuration 2, which restricted training to a limited set of models, in this case, Logistic Regression, Random Forest, and XGBoost. H2O AutoML still led with the longest time at 804.16 seconds. In the subsequent comparison of memory usage, H2O AutoML consumed the most, at 458.20 MB, out of all the frameworks. AutoGluon had 30.12 MB, followed by TPOT and PyCaret with 92.24 MB and 70.98 MB, respectively. The complexity of the training process is the reason why it required the most memory when compared to other frameworks. Also, the fact that H2O AutoML has more available algorithms than AutoGluon is better suited for rapid model implementation. With an average of 58.55% and 57.85% in Configurations 1 and 2, respectively, it is demonstrated that H2O AutoML requires higher CPU usage because of the complexity of the algorithm process.

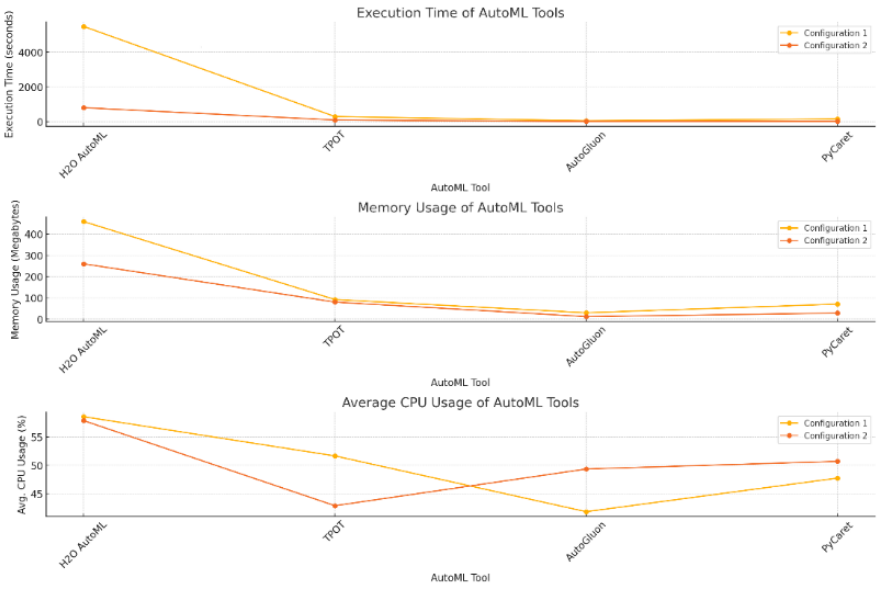

Figure 2: Model diversity resource usage and training time.

Experiment 2: Cross-Validation Fold

In Experiment 2 as shown in **Figure 3**, all of the configurations, which represented higher levels of complexity and resource demand, showed clear differences in how efficiently the AutoML frameworks used computing power. In Configuration 1 (3-fold), H2O AutoML already used a lot of resources, taking over 3200 seconds and 362MB of memory. On the other hand, PyCaret used the least amount of time because its architecture is lightweight. In Configuration 2 (5-fold), all frameworks were used more, but H2O AutoML time and CPU usage went up to 4884 seconds, with CPU usage at about 99%, while PyCaret memory usage only went up by 40MB. H2O AutoML reached its peak at Configuration 3 (10-fold), with 11159 seconds and 637MB of usage. This showed that the CPU was fully used at 98.85%. TPOT also scaled up a lot, reaching 660 seconds and 164MB. AutoGluon went up in a way that made sense (307 seconds, 99MB), but it stayed within a reasonable range. PyCaret was able to stay efficient even at the highest setting, which was 56 seconds and 55MB, and it did not use up too many resources. These results show that PyCaret is best for situations where there are not many resources or where things need to be done quickly. H2O AutoML is best for situations where there are enough resources and a lot of exploration needs to be done.

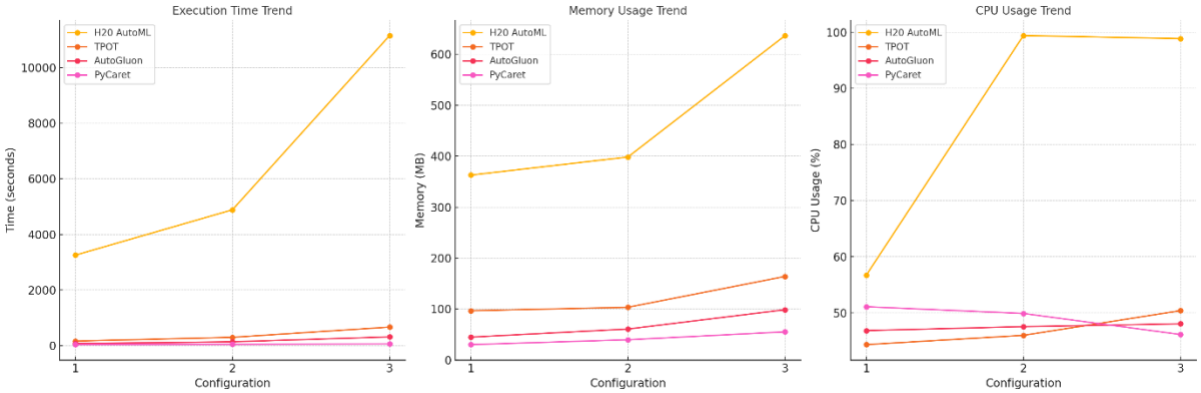

Figure 3: Cross-validation fold resources usage and training time.

Experiment 3: Number of Model

(1) All models in framework

In Experiment 3 as shown in **Figure 4** for Configuration 1, each of the AutoML frameworks showed clear differences in how well they worked across three setups. In Configuration 1, PyCaret was still the fastest, finishing in 65.21 seconds with only 50MB of memory and moderate CPU usage (54.5%). AutoGluon and TPOT took longer (222.19s and 307.31s) and needed more memory. H2O AutoML used a lot of resources, taking more than 4,800 seconds and almost 490MB of memory, with a high CPU usage of 94.25%. When switched to Configuration 2, all frameworks used more resources. H2O AutoML went up to 10,706 seconds with 590MB of memory and a high CPU load (77.95%). TPOT went up to 427.5 seconds and showed a spike in CPU usage (89.9%). AutoGluon and PyCaret remained stable. PyCaret time only went up a little bit (67.29s), and its memory stayed below 60MB. In Configuration 3, H2O AutoML used the most resources, with over 18,000 seconds, 844MB of memory, and 97% of the CPU. TPOT and AutoGluon also took longer and used more memory, but not as much. Interestingly, PyCaret execution time went down to 54.52 seconds, even though the configuration was heavier. It also kept its low memory usage (58.74MB) and balanced CPU usage (55.5%). In general, the results show that PyCaret is the most stable and resource-efficient, AutoGluon is well-balanced for moderate setups, TPOT uses more CPU power as the load increases, and H2O AutoML, while thorough, takes a lot more time and system resources as the workload increases.

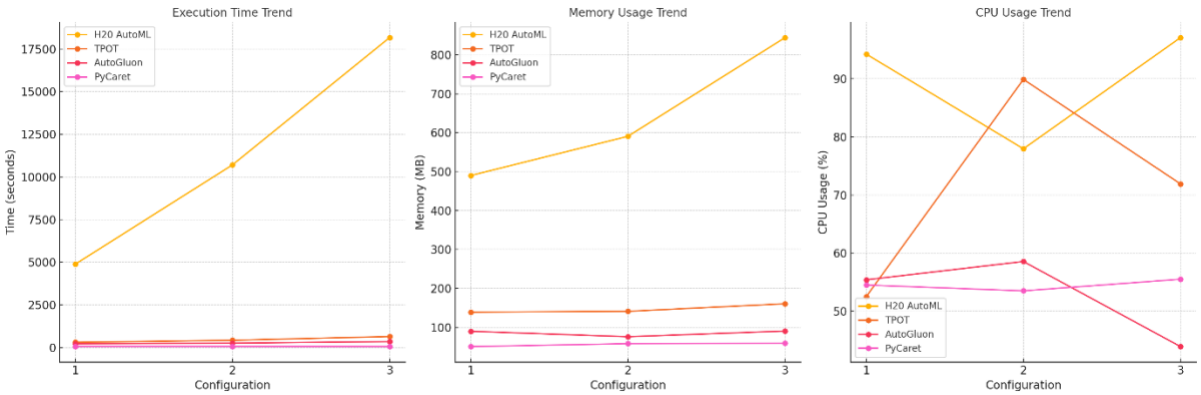

Figure 4: Number of model resource usage and training time using all model in AutoML framework.

(2) Specific model (Random Forest, Logistic Regression, XGBoost)

As shown in **Figure 5** above, in Experiment 3 the AutoML frameworks showed different levels of computational efficiency across all three configurations, which showed higher levels of workload. In Configuration 1, PyCaret stood out as the lightest low-code machine learning tool. It finished in just 12.57 seconds, used the least memory (15MB), and had moderate CPU usage (50.55%). AutoGluon used a little more CPU (61.3%), but it was still very fast (27.37 seconds) and used memory well. TPOT took longer (80.89 seconds) and used more memory (34.74MB), while H2O AutoML already used a lot (431.91 seconds, 200MB). As it is switched to Configuration 2, PyCaret kept its low execution time (14.19 seconds) and low memory use (18.83MB). AutoGluon was close behind, and it even improved its execution time (17.11 seconds). Both TPOT and H2O AutoML used a little more time and memory. When using Configuration 3, which was the hardest, H2O AutoML took a lot longer to run (1,557.87 seconds) and used almost all of the CPU (97.35%). TPOT also took longer (113.05 seconds) and used 48MB of memory. AutoGluon stayed stable, with a small-time increase (22.40 seconds) and low memory usage. PyCaret, on the other hand, showed great consistency, with very little time (14.81 seconds) and memory usage (18.15MB). These results show that PyCaret is the most resource-efficient and consistent framework across all configurations. AutoGluon strikes a good balance between speed and CPU efficiency, TPOT is average and scales well, and H2O AutoML is the most computationally demanding, especially when used with heavier configurations.

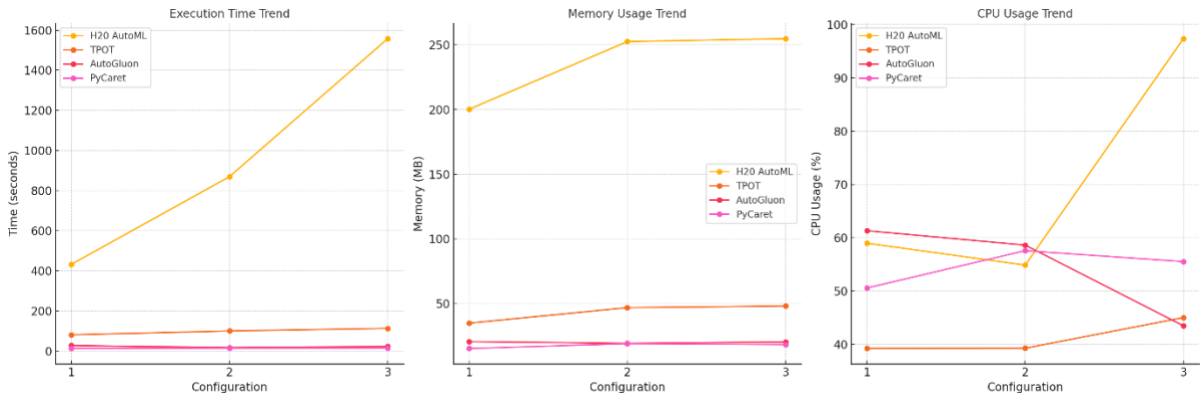

Figure 5: Number of model resource usage and training time using specific model in AutoML framework.

Overall, In Experiment 1, H2O AutoML had the longest execution times overall, reaching a peak of 5483.19 seconds in Configuration 1. It also consumed substantial memory (458.20 MB) and had an average CPU usage of 58.55%, which shows that it is resource intensive. On the other hand, AutoGluon was very efficient, finishing the same configuration in only 51.05 seconds while using only 30.12 MB of memory and 41.85% of the CPU, making it the fastest and most resource-efficient choice. Experiment 2 further demonstrated the variability of H2O AutoML performance. In Configuration 2, execution times reached 4884.37 seconds, while TPOT and AutoGluon kept their execution times lower at 291.76 seconds and 130.37 seconds, respectively. In Experiment 3, H2O AutoML took the longest time to run, at 18168.55 seconds, which shows how much computational resources it needs. TPOT and AutoGluon, on the other hand, stayed more stable and efficient. In summary, while H2O AutoML delivers strong predictive performance, its high resource consumption limits its suitability in constrained environments. AutoGluon and PyCaret offer more efficient alternatives suited to rapid prototyping, though they may be less effective for large-scale or complex tasks.

3.6 Model Overfitting and Generalisation

A key concern in any machine learning study is whether the trained models generalize well to unseen data or whether they overfit to the training set. Several measures were taken in this study to mitigate overfitting. First, the dataset was partitioned into three subsets: 70% for training, 15% for validation, and 15% for testing. The validation set was used exclusively for hyperparameter tuning, while the held-out test set was used only for final performance evaluation, ensuring that no information from the test set leaked into the model training process. Second, cross-validation was applied in Experiment 2, with fold sizes of 3, 5, and 10. The consistent performance observed across all fold sizes, particularly the stable accuracy and F1-Score of H2O AutoML across 3-fold, 5-fold, and 10-fold settings which indicates that the models generalize well rather than memorizing training data. A significant drop in performance with higher fold counts would have suggested overfitting; however, the results show that performance either improved slightly or remained stable, which is a strong indicator of robust generalization. Third, the use of ensemble methods (particularly in H2O AutoML and AutoGluon) inherently reduces overfitting risk by combining multiple base learners whose errors tend to cancel out. TPOT's use of genetic programming similarly evolves diverse pipeline configurations, reducing the likelihood of relying on a single overfit model. These combined strategies provide reasonable assurance that the reported performance metrics reflect true generalization ability rather than overfitting to the training data.

3.7 Operationalisation in Malaysian Academic Settings

The dataset for this study was sourced from the Polytechnic Institute of Portalegre, Portugal. Models trained on this dataset cannot be directly applied to contexts in other countries, including institutions of higher education in Malaysia. While the two regions differ in their institutional system structures, grading frameworks, student demographic profiles, and socio-economic factors, differences that bar the original model from direct deployment; however, the methodological framework of this study is transferable. It can guide Malaysia to build an automated machine learning-based prediction system. Features such as age at enrolment, gender, marital status, and international student status can be reused directly. Most of the remaining features can also be mapped to corresponding data already collected locally. The difficulty of adaptation is far lower than initially expected, so cross-regional application of this approach is fully feasible.

To adapt this study to the local research context of Malaysia, localization adjustments must be made to the dataset features of the original Portuguese study. This paper first outlines that partial replacement is required for socio-economic features, then specifies adaptation rules across three categories: demographic features are mapped to the Malaysian Standard Classification of Occupations (MASCO) [24] and the Malaysian Qualifications Framework (MQF) [25] levels. The financial features related to financial aid are aligned with local official programs such as PTPTN loan recipient status, JPA scholarships, or university bursaries. The macroeconomic features adopt the unemployment rate, inflation rate, and GDP data released by the Department of Statistics Malaysia (DOSM).

Among the standard student academic measurement variables uniformly recorded by all higher education institutions in Malaysia, the original admission score attributes must be replaced with four types of local admission qualification scores such as SPM, STPM, Matriculation GPA, and Diploma CGPA, in line with the specific requirements of each institution and its programs. Three core adaptation tasks must be completed to achieve this: remapping occupational and educational classification codes to align with Malaysia's national standards, replacing admission indicators, and calibrating macro-level indicators. This implementation can be advanced through the support of public polytechnics and community colleges.

3.8 Feature Importance and Model Interpretability

In research on student academic risk prediction, model interpretability must be treated as a core requirement rather than an auxiliary consideration added as an afterthought. Models that simply output "at risk" or "not at risk" labels hold limited practical value for frontline educators, who can only design and implement targeted intervention plans once they understand the reasoning behind these classifications. This study addresses that need directly by examining the top-performing models generated by AutoML frameworks through a feature importance analysis.

Ranked by overall influence as shown in **Figure 6**, the predictors fall into four broad categories: academic performance factors, particularly curricular units approved and grades in the first and second semesters, emerged as the most influential, followed by socio-economic factors such as tuition fee payment status, demographic factors including age at enrolment, and finally macroeconomic factors such as GDP and unemployment rate, which carried the least weight. Each framework also lends itself to a different interpretability approach: H2O AutoML is particularly accessible for education stakeholders without a technical background, PyCaret offers a built-in feature importance function based on SHAP values, and TPOT allows users to extract feature importance directly from its exported scikit-learn pipeline components. Taken together, these findings suggest that AutoML, when paired with the right interpretability approach, can deliver not only accurate predictions but also practical insights that educational institutions can apply to support targeted academic interventions.

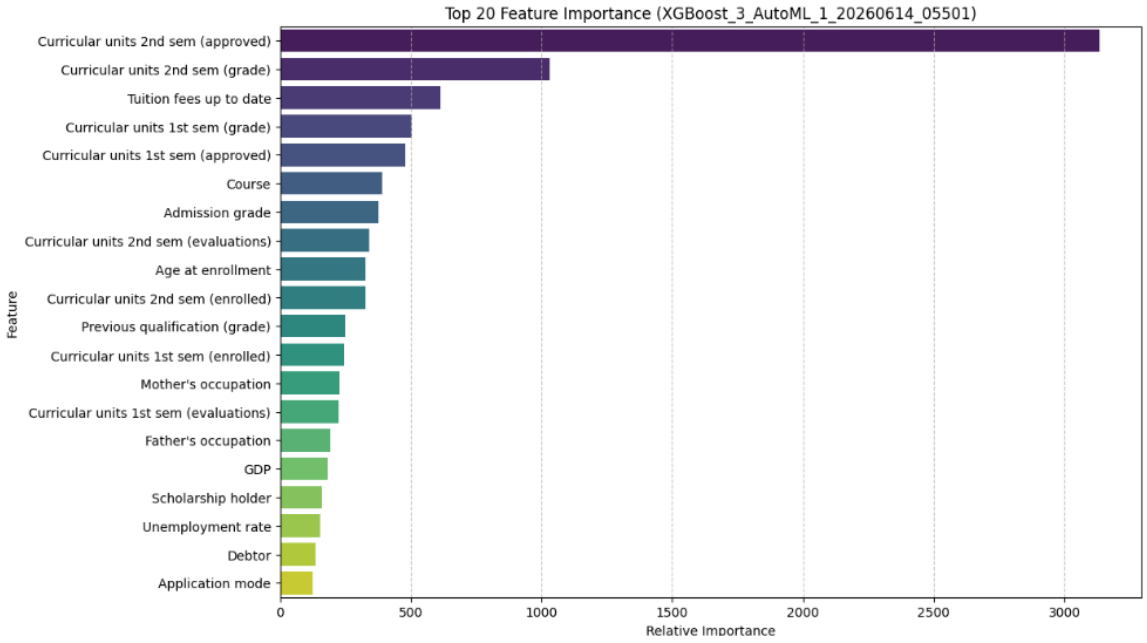

Figure 6: Feature importance analysis

4. CONCLUSIONS

This study has explored the AutoML techniques for predicting student dropout and academic success in higher education, which highlight the potential of these frameworks to enhance the higher education system. H2O AutoML, TPOT, AutoGluon and PyCaret show notable differences in terms of the model performance, runtime, memory usage, and CPU efficiency. These findings can provide valuable insight into selecting suitable frameworks based on resource constraints and analysis needed in educational settings. However, some key limitations of this study are that it only utilized one dataset which may limit the generalizability of the findings.

Future research should consider expanding the dataset by including a broader range of higher education institutions, capturing diverse academic structures, regional contexts, and student populations. As AI-driven predictive tools are increasingly adopted by educational institutions, the need for transparent and interpretable systems has become increasingly important. Moving forward, the development of AutoML should not only prioritize accuracy but also provide insight into factors that influence student outcomes. Techniques such as feature importance analysis and explainable AI can provide valuable insight into these problems. Additionally, by deploying these models in real-world educational environments would help evaluate the effectiveness of early interventions and their impact on academic performance and equity across student groups.

ACKNOWLEDGEMENT

No funding was received for this study.

REFERENCES

[1] V. Realinho, J. Machado, L. Baptista, M. V. Martins, "Predicting student dropout and academic success," vol. 7, no. 11, p. 146, 2022.

[2] D. Andrade-Girón, J. Sandivar-Rosas, W. J. Marin-Rodriguez, E. Susanibar-Ramirez, E. Toro-Dextre, J. Ausejo Sánchez, H. Villarreal-Torres, J. Angeles-Morales, "Predicting Student Dropout based on Machine Learning and Deep Learning: A Systematic Review," *ICST Transactions on Scalable Information Systems,* 2023.

[3] C. Cho, Y. Yu, H. Kim, "A Study on Dropout Prediction for University Students Using Machine Learning," *Applied Sciences,* vol. 13, p. 12004, 2023.

[4] D. Dake, C. Buabeng-Andoh, "Using Machine Learning Techniques to Predict Learner Drop-out Rate in Higher Educational Institutions," *Mobile Information Systems,* vol. 2022, pp. 1-9, 2022.

[5] A. Asselman, M. Khaldi, A. Souhaib, "Enhancing the prediction of student performance based on the machine learning XGBoost algorithm," *Interactive Learning Environments,* vol. 31, pp. 1-20, 2021.

[6] A. Althibyani, "Predictive Modeling of Dropout in MOOCs Using Machine Learning Techniques," *International Journal of Intelligent Systems and Applications in Engineering,* vol. 12, no. 2, p. 247–256, 2024.

[7] F. Moghimi, M. Metzger, "Predicting student dropouts via machine learning," *SSRN,* 2022.

[8] A. Alaiad, A. Migdady, R. M. Al-Khatib, O. Alzoubi, R. A. Zitar, L. Abualigah, "Autokeras Approach: A Robust Automated Deep Learning Network for Diagnosis Disease Cases in Medical Images," *Journal of Imaging,* vol. 9, no. 3, p. 64, 2023.

[9] D. Fattah, A. Ahmed, A. S. Desuky, M. Zaki, "AutoKeras and particle swarm optimization to predict the price trend of stock exchange," *Bulletin of Electrical Engineering and Informatics,* vol. 11, pp. 1100-1109, 2022.

[10] H. A. Madni, M. Umer, A. Ishaq, N. Abuzinadah, O. Saidani, S. Alsubai, M. Hamdi, I. Ashraf, "Water-quality prediction based on H2O AutoML and explainable AI techniques," *Water,* vol. 15, no. 3, p. 475, 2023.

[11] Y. Mnyawami, H. Maziku, J. Mushi, "Enhanced Model for Predicting Student Dropouts in Developing Countries Using Automated Machine Learning Approach: A Case of Tanzanian's Secondary Schools," *Applied Artificial Intelligence,* vol. 36, 2022.

[12] M. Vaarma, H. Li, "Predicting student dropouts with machine learning: An empirical study in Finnish higher education," *Technology in Society,* vol. 76, p. 102474, 2024.

[13] M. Shi, W. Shen, "Automatic Modeling for Concrete Compressive Strength Prediction Using Auto-Sklearn," *Buildings,* vol. 12, p. 1406, 2022.

[14] P. Ribeiro, A. Saini, J. Moran, N. Matsumoto, H. Choi, M. Hernandez, and J. H. Moore, "TPOT2: A New Graph-Based Implementation of the Tree-Based Pipeline Optimization Tool for Automated Machine Learning," 2024.

[15] J. Niyogisubizo, L. Liao, E. Nziyumva, E. Murwanashyaka, P. C. Nshimyumukiza, "Predicting student's dropout in university classes using two-layer ensemble machine learning approach: A novel stacked generalization," *Comput. Educ. Artif. Intell.,* vol. 3, p. 100066, 2022.

[16] E. Osemwegie, F. Amadin, "Student Dropout Prediction Using Machine Learning," *FUDMA Journal of Sciences,* vol. 7, pp. 347-353, 2023.

[17] H. Park, S. Yoo, "Early Dropout Prediction in Online Learning of University using Machine Learning," *JOIV: International Journal on Informatics Visualization,* vol. 5, p. 347, 2021.

[18] "AutoML," [Online]. Available: https://www.automl.org/automl/. [Accessed 15 April 2025].

[19] "H2O.ai," 2016-2025. [Online]. Available: https://docs.h2o.ai/h2o/latest-stable/h2o-docs/index.html. [Accessed 15 April 2025].

[20] "AutoML: information about automated machine learning," [Online]. Available: https://automl.info/tpot/. [Accessed 15 4 2025].

[21] "Auto Gluon AutoML," 2025. [Online]. Available: https://auto.gluon.ai/stable/index.html. [Accessed 15 April 2025].

[22] "PyCaret 3.0," [Online]. Available: https://pycaret.gitbook.io/docs. [Accessed 15 April 2025].

[23] L.-S. Sweet, C. Müller, M. Schulz, and J. Franke, "Cross-Validation Strategy Impacts the Performance and Interpretation of Machine Learning Models," *Artificial Intelligence for the Earth Systems,* vol. 2, no. 4, 2023.

[24] D. o. S. Malaysia, "Malaysian Standard Classification of Occupations (MASCO)," Department of Statistics Malaysia, Putrajaya, 2020.

[25] M. Q. Agency, "Malaysian Qualification Framework (MQF) 2nd Edition (Updated)," MQA, 2024.

[26] H. A. Madni, M. Umer, A. Ishaq, N. Abuzinadah, O. Saidani, S. Alsubai, M. Hamdi, I. Ashraf, "Water-quality prediction based on H2O AutoML and explainable AI techniques," vol. 15, no. 3, p. 475, 2023.