Scalable and Efficient Student Behavior Prediction using Parallelized Clustering and AHP-weighted KNN

Li Guozhang[1], Rayner Alfred[2,*], Rayner Pailus[2], Xu Fengchang[3] & Haviluddin[4]

[1]College of Information Engineering, Hainan Vocational University of Science and Technology, Haikou 571126, Hainan, China.
[2]Faculty of Computing and Informatics, Universiti Malaysia Sabah, Jalan UMS, 88400 Kota Kinabalu, Sabah, Malaysia.
[3]Shandong Light Industry Vocational College, Zibo City, Shandong Province, China.
[4]Faculty of Computer Science and Information Technology, Universitas Mulawarman, Jalan Sambaliung, Kampus Gunung Kelua, Samarinda, East Kalimantan, Indonesia
*E-mail: ralfred@ums.edu.my

Abstract. This study proposes a scalable and efficient approach for predicting student behaviour in large-scale educational environments. It introduces a parallelized hybrid model that combines Density-Based Optimized K-Means clustering, Analytic Hierarchy Process (AHP) feature weighting, and Hierarchical K-Nearest Neighbours (KNN), implemented using Apache Spark. The main research question is how to improve scalability, accuracy, and computational efficiency of student behaviour prediction when dealing with large, complex datasets. The model addresses key limitations of traditional methods, such as handling heterogeneous data, treating all features equally, and high computational cost. Two main innovations are presented. First, AHP is used to assign structured importance to features, allowing critical factors like attendance and study time to have greater influence on prediction accuracy. Second, clustering and prediction are parallelized using Spark, enabling efficient real-time processing of large datasets. The approach was evaluated using 18,586 student records and more than 20 million behavioural entries. Results show that Hierarchical KNN consistently outperforms standard KNN as dataset size increases. While traditional KNN shows unstable error rates, peaking at 9.4%, Hierarchical KNN maintains lower and more stable errors between 5.16% and 6.08%. Execution time was also significantly reduced through parallel processing, though gains were limited by communication overhead. Overall, the proposed model offers a robust framework for real-time behaviour analysis, academic risk detection, and targeted educational intervention.

Keywords: *analytic hierarchy process (ahp); density-based optimized k-means; educational data mining; feature weighting; hybrid model; parallelized feature selection; parallelized model training; student behavior prediction.*

1 Introduction

The increased enrolment in higher education institutions highlights the need for more sophisticated data mining approaches that can handle big data, including

Received October 15th, 2024, 1st Revision July 14th, 2025, 2nd Revision August 15th, 2025, 3rd Revision September 15th, 2025, Accepted for publication October 9th, 2025.

student behaviour analytics [1]. Traditional data management systems offer basic statistical insights, but they lack the depth required for comprehensive student profiling and performance prediction [2]. Consequently, modern approaches that utilize clustering algorithms integrated with machine learning and big data analytics are necessary to provide accurate and efficient results. These technologies offer real-time, data-driven insights into student behaviour, which can be used to improve teaching methods, identify students at risk of failure, and foster a more supportive learning environment [3][4].

In predictive modelling, particularly in educational data mining, feature importance plays a crucial role in improving the accuracy of predictions. When certain features (such as study time, participation, and exam performance) are more predictive of student behaviour, placing appropriate weight on these features can significantly enhance the model's effectiveness. For instance, K-Nearest Neighbours (KNN), a commonly used algorithm for student behaviour prediction, is sensitive to the scale and importance of input features. If all features are treated equally, the model may be influenced by irrelevant or less important features, leading to suboptimal results. Therefore, feature weighting is a critical step to ensure that the most relevant factors receive more influence in the distance calculations that KNN relies on for prediction [5]-[7].

In current literature, Analytic Hierarchy Process (AHP) has been applied in multi-criteria decision-making scenarios and has shown promise in improving model performance by providing more accurate feature weighting. However, limited studies have specifically examined the impact of AHP feature weighting on student behaviour prediction models such as KNN, making this research question timely and relevant [8]. AHP offers the advantage of considering human expertise and domain knowledge in assigning feature importance. Given its multi-criteria decision-making nature, AHP could provide a more robust way to rank student behaviour features in educational datasets, resulting in improved predictive accuracy when integrated into KNN models [9].

There are several hybrid models involving clustering, feature selection, and KNN in various applications. For instance, KNN was integrated with Fuzzy C-Means (FCM) clustering to improve prediction accuracy, particularly in classification tasks [10]. While similar in using feature weighting and KNN, the distinction lies in the AHP-based feature weighting in your model, which offers a systematic decision-making approach for prioritizing features, and no parallelization approach applied in these works. Another related work is the development of a novel parallel hybrid model based on series hybrid models of ARIMA and ANN Models [11]. The ARIMA-ANN hybrid models are primarily time-series prediction models, where ARIMA handles linear patterns and ANN captures non-linear trends. These models benefit from parallelization mainly in the ANN

component, which can use parallel computing during backpropagation and model training. However, ARIMA itself is a sequential model, limiting the overall parallel efficiency. Parallelism is mainly achieved by running both models (ARIMA and ANN) in parallel, but it is less flexible compared to the multi-stage parallelism in the model proposed in this paper.

A hybrid model was also introduced for traffic flow prediction integrates KNN with multiple clustering algorithms to handle varying feature distributions across regions [12]. The focus is on optimizing KNN performance through better clustering, though the feature weighting mechanism differs, as this model does not use AHP and parallelization. Another hybrid model was also proposed for heart disease prediction using a combination of unsupervised clustering and supervised learning [13]. This model employs collaborative clustering, where multiple clustering algorithms share information to enhance accuracy, combined with ensemble learning for final predictions. Although this model focuses on a different domain (heart disease prediction), its hybrid nature and the use of multiple clustering techniques for behaviour classification align with the proposed approach in this paper, albeit without the application of AHP-based feature weighting for feature selection [34] and the application of parallelization to improve the execution efficiency of the hybrid model [14].

Some models combine KNN with LSTM for trajectory prediction, leveraging KNN for high-density data and LSTM for time series data with low-density points [15]. While trajectory prediction is different in scope, the idea of hybridizing different methods (KNN and clustering) resonates with the proposed model's design in this work. However, in this work, the proposed model emphasis on parallelization for execution efficiency of the proposed hybrid model and the application of AHP for feature weighting in behaviour prediction offers a more tailored approach for feature relevance in behavioural data.

In summary, while similar hybrid models exist, the proposed model in this work stands out by integrating the AHP method for feature weighting, adding a structured decision-making process to feature importance and the application of parallelization for efficient execution, which are not explored thoroughly in other hybrid KNN models. The use of hybrid models (e.g., combining clustering, feature weighting, and classification techniques) has been shown to outperform single method approaches in predicting student success. However, current research lacks sufficient exploration of AHP’s role in feature weighting, particularly in the context of KNN models for educational applications [16]. Current student behaviour prediction methods face significant challenges, particularly with traditional clustering techniques like K-Means, which struggle to accurately segment students due to their inability to manage varying densities in engagement and learning patterns, resulting in less meaningful and

representative clusters. Additionally, many prediction models fail to optimize accuracy by treating all features equally, neglecting structured approaches like the Analytic Hierarchy Process (AHP), which can prioritize the most relevant features for models such as K-Nearest Neighbours (KNN). However, as the volume of educational data grows, the computational demands of algorithms like K-Means and KNN become a major bottleneck, hindering real-time analysis and decision-making. This underscores the critical need for parallelization to improve both the efficiency and scalability of these algorithms, enabling them to process large educational datasets rapidly and support real-time student behaviour predictions in dynamic learning environments. By leveraging parallel computing frameworks, such as Apache Spark, the performance of these algorithms can be significantly enhanced, making them more suitable for large-scale educational data mining applications.

The purpose of this paper is to develop and evaluate a parallelized hybrid model that integrates Density-Based Optimized K-Means clustering, Analytic Hierarchy Process (AHP)-based feature weighting, and K-Nearest Neighbours (KNN) for predicting student behaviour based on the running or execution time. In other words, the study aims to evaluate the overall performance of the proposed parallelized hybrid model, which combines clustering, AHP-based feature weighting, and KNN, for efficient and accurate student behaviour prediction, particularly when applied to large-scale educational datasets. By implementing parallelization techniques, the paper seeks to address scalability challenges and improve the real-time processing capability of the model in dynamic educational environments.

2 Methods

This study uses a quantitative experimental design to develop and evaluate clustering and predictive models for student behavior analysis. It combines descriptive analytics, using clustering to segment students by engagement patterns, with predictive modeling to forecast academic success or risk. By integrating both approaches, the study offers deeper insights into student behaviors. To ensure scalability, the models are implemented using Apache Spark for parallel processing, enabling efficient handling of large educational datasets. This approach enhances processing speed and supports real-time data analysis, making it suitable for large-scale learning environments and practical deployment in educational systems. The methodology involves a multi-stage machine learning pipeline that integrates three key components:

Density-Based Optimized K-Means Clustering: This algorithm is used to segment students into meaningful clusters based on their engagement levels and learning patterns. It improves upon traditional K-Means by handling varying

densities in student behavior, resulting in more representative and compact clusters.

Analytic Hierarchy Process (AHP) for Feature Weighting: AHP is applied to prioritize the most relevant features in the data, ensuring that features critical to predicting student behavior receive appropriate weighting. This structured feature weighting is integrated into the predictive modelling stage to enhance the accuracy of predictions.

K-Nearest Neighbours (KNN) for Prediction: KNN is employed to predict student behaviour by identifying similar students based on the weighted features. The inclusion of AHP helps improve the accuracy of KNN by ensuring that more relevant features have a greater influence on the prediction process.

The integration of these techniques forms the foundation of the proposed hybrid model. A key innovation of this study is the parallelization of clustering and KNN algorithms, allowing for efficient processing of large datasets in real-time or near-real-time environments. This is achieved using the Spark platform, which distributes computational tasks across multiple nodes, significantly improving scalability and reducing execution time. The research methodology also places a strong emphasis on data preprocessing, model evaluation, and performance optimization. Each stage of the pipeline is rigorously tested to ensure that the models achieve optimal performance in terms of both clustering accuracy and predictive power. This includes evaluating the compactness and separation of the clusters formed by the Density-Based Optimized K-Means algorithm and assessing the accuracy and efficiency of the AHP-weighted KNN predictions. Figure 1 illustrates the flowchart that describes the key phases of the proposed research that will be executed in this thesis. All three research objectives are also mapped into the research methodology to ensure all of them are covered and achieved successfully.

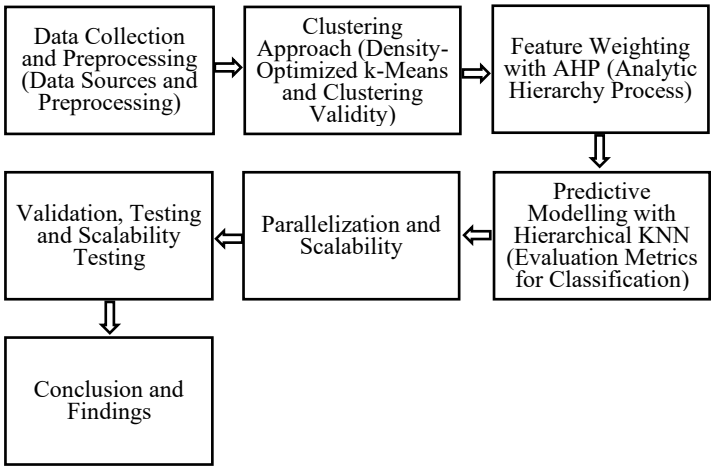

Figure 1 Flowchart illustrating the key phases of the proposed research.

2.1 Data Sources

The dataset for this study was collected from Chinese universities and colleges, covering student behavior indicators such as library visits, physical activity, and consumption patterns. The data, extracted from the university's digital campus infrastructure, spans the period from March 2015 to March 2017, and includes 18,586 student records. These records include consumption data (8,332,810 entries), library book borrowing (1,568,347 entries), attendance records (8,595,864 entries), and library access logs (2,370,988 entries). The dataset also contains academic performance, physical exercise records, and wireless network access logs, making it suitable for a comprehensive analysis of student behavior [16][17]. Two main datasets are used call the consumption patterns and the learning efforts.

2.2 Data Collection and Preprocessing

The original dataset was subjected to preprocessing steps, including handling missing values through techniques like mean/mode imputation and normalization of numerical data such as study hours and spending using the min-max normalization method (Eq. (1)). Categorical data, like student demographics, were one-hot encoded to convert them into a machine-readable format. These preprocessing steps ensured that the data were suitable for clustering and predictive models [16].

$$x' = \frac{|x - min|}{\max - min} \tag{1}$$

2.3 Machine Learning Algorithm – Hierarchical KNN

In this study, the term *Hierarchical KNN* refers to a two-level or multi-stage enhancement of the traditional KNN algorithm, designed to improve scalability and prediction accuracy in large datasets. Unlike standard KNN, which calculates distances across the entire dataset uniformly, the Hierarchical KNN implementation first performs a pre-clustering step (Modified Optimized Density-based clustering) to group similar data points. Then, Analytic Hierarchy Process (AHP)-based feature weighting is used to prioritize the most relevant features in the data, ensuring that features critical to predicting student behavior receive appropriate weighting and finally, KNN is applied within or across selected clusters, thereby narrowing the search space and reducing computational complexity. This hierarchical structure introduces a layered decision process:

1. Macro-level grouping (clustering) to localize similar behaviour patterns.
2. AHP-based feature weights integration that will significantly improve the prediction accuracy for identifying student behaviours.

In the context of Spark, this structure also aligns with the parallel computing architecture, where clustering and neighbour search can be independently distributed across nodes. As a result, this hierarchical approach improves both the efficiency and accuracy of the prediction model, particularly when working with high-dimensional or large-scale student behaviour datasets.

2.3.1 Macro-level Grouping (Clustering)

Two clustering approaches were implemented, Traditional K-Means and Density-Based Optimized K-Means. The clustering results were validated by assessing intra-cluster compactness and inter-cluster separation using evaluation metrics like cluster validity index ($V(k)$) for k clusters, where intra-class similarity is represented by the average distance between each sample point within each cluster and its corresponding cluster centroid, as shown in Eq. (2).

$$\boldsymbol{V(k)} = \frac{\boldsymbol{D_{out} - D_{in}}}{\boldsymbol{D_{out} + D_{in}}} \tag{2}$$

$$D_{in} = \frac{1}{N}\sum_{i=1}^{k}\sum_{j=1}^{m} D\left(x_j, p_i\right) \tag{3}$$

In Eq. (3), D(xj, pi) represents the distance between sample point xj, and the centroid pi of its corresponding cluster. The dissimilarity between clusters measures the degree of separation between different clusters and is expressed as the average distance between cluster centers, as shown in Eq. (4), where $D(p_i, p_j)$ represents the distance between centroid p_i and centroid p_j of two different clusters.

$$D_{out} = \frac{1}{k}\sum_{i,j=1}^{k} D(p_i, p_j) \quad (4)$$

A Hierarchical KNN model is then developed and implemented to predict student behavior based on the clusters and weighted features. Predictions were evaluated using Relative Error (RE) (See Eq. (5)) and Standard Error (SE) (See Eq. (6)) [3].

$$RE = \frac{Number\ of\ Incorrect\ Predictions}{Total\ number\ of\ prediction} \times 100 \quad (5)$$

$$SE = \frac{sd}{\sqrt{n}} \quad (6)$$

The Analytic Hierarchy Process (AHP) is applied to prioritize the most relevant features for the predictive model. Using a pairwise comparison matrix, features like attendance, academic performance, and study time were weighted according to their importance in predicting student behavior.

In this study, the AHP pairwise comparison matrix was constructed using five primary behavioural features: attendance, study time, academic performance, library access, and consumption pattern. Expert input from faculty members was used to establish the relative importance of each feature in predicting student behaviour. The comparison was based on Saaty's 1–9 scale, where 1 indicates equal importance and 9 indicates extreme importance of one feature over another.

The resulting normalized weights were as follows: Attendance: 0.32, Study Time: 0.27, Academic Performance: 0.2, Library Access: 0.12, Consumption Pattern: 0.08.

These weights indicate that attendance and study time are the most influential features, aligning with domain knowledge that consistent class participation and dedicated study hours strongly impact academic outcomes. The Consistency Ratio (CR) was calculated at 0.07 (< 0.1), confirming acceptable consistency in expert judgments. These AHP-derived weights were then applied to the feature set before KNN distance calculations, ensuring that more predictive features had proportionally greater influence in the similarity assessment.

This weighting improved the accuracy of the subsequent KNN model by emphasizing the most relevant features [9]. The hierarchical KNN model integrated the AHP-based feature weights, significantly improving the prediction accuracy for identifying student behaviors.

2.3.2 AHP-based Feature Weights Integration with KNN

The produced cluster results are then used as the class label for predicting the student's behavior. Before performing the predictions, the Analytic Hierarchy

Process (AHP) model is applied to the Learning Efforts and Consumption Patterns features compute produce the weighted features. These weighted features and class labels are combined to create an Updated Dataset. Finally, the updated dataset, containing both class labels and feature weights, is used to predict student behavior using a Hierarchical K-NN (K-Nearest Neighbors) algorithm.

It is important to clarify that the term *Hierarchical KNN* in this study does not refer to a tree-based nearest neighbor search structure, as sometimes used in literature. Instead, the 'hierarchical' aspect arises from the multi-stage pipeline designed in this work, which integrates (1) clustering through the Density-Based Optimized K-Means algorithm, (2) Analytic Hierarchy Process (AHP)-based feature weighting, and (3) K-Nearest Neighbours (KNN) prediction. This layered sequence of clustering, weighting, and classification is what constitutes the hierarchical design in our approach, ensuring improved scalability and accuracy for large-scale educational datasets.

In summary, the process starts with student data, applies clustering and AHP to generate class labels and weighted features, and ultimately predicts student behavior using a hierarchical K-NN model. Based on the cluster label produced using the modified density-based optimize *k*-means clustering, the similarity between the target student and the existing students is calculated based on the updated student features. To improve the accuracy of the prediction, the data is standardized before calculating the similarity between two students' behaviors. Subsequently, *k* students having the most similar characteristics with the target student *x* are identified based on their similarity and the predicted class for the target student *x* is determined by the majority of labels in the K students. The similarity between the target student and the K students in the training sample is reflected by the weighted Euclidean distance.

2.3.3 Parallelization

In this stage, the task of implementing the Density-Based Optimized K-Means and Hierarchical KNN models in parallel on the Spark cluster is performed. The performance of the parallelization approach is measured based on the execution time and speedup ratio as the number of worker nodes and the dataset size increase.

2.3.3.1 Parallelization Implementation of Density-based Optimized K-Means Algorithm

Given the substantial volume of data involved in student clustering for this study, and to ensure the system's scalability for processing larger datasets in the future, the algorithm was designed and implemented in a parallelized manner on Spark

[19]. The parallel implementation of the Density-Based Optimized K-means algorithm, based on student behavior characteristics, is illustrated in Figure 2. Notably, the parallel implementation of density-based optimized K-means clustering is divided into two stages. In the first stage, students who meet the density requirements are clustered based on the density of all students that have been scanned. In the second stage, all students are scanned to assign students to a cluster and update the cluster center points. The task of *Submission & Data Loading Process* is the entry point for the entire parallel program.

Task submission & Data Loading: Apache Spark is a unified computing engine for parallel data processing, widely used in big data analytics and supporting languages like Python, SQL, and machine learning tasks. In this study, Spark was used for data submission, clustering, and analysis on a local computer [14]. Execution began with the spark-submit command, specifying the Python application file and configurations such as driver memory, executor memory, deploy mode, and core usage. This enabled efficient processing of large datasets using pre-written Spark applications.

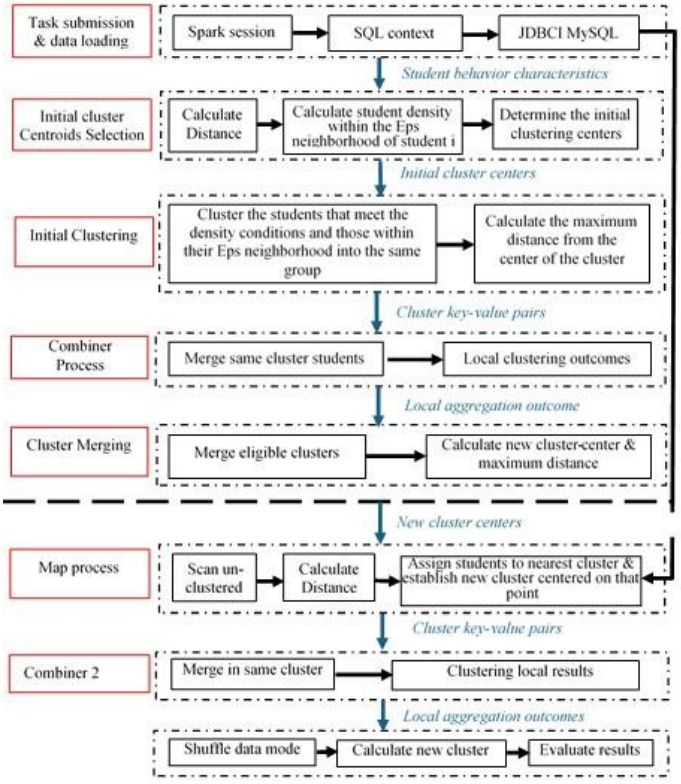

Figure 2 Parallelization of Optimized K-means Algorithm Based on Spark.

Initial Cluster Centroids Selection: The initial k centroids are randomly selected since true cluster centers are unknown. Each data point is then assigned to the nearest centroid based on Euclidean distance. Centroids are updated by averaging the data points in each cluster, and the process repeats until the centroids stabilize, typically after several iterations. However, random initialization can lead to high errors and poor clustering accuracy. To address this, the initial centroid selection is optimized using a density-based approach, which improves starting positions and reduces the need for excessive iterations, ultimately enhancing clustering performance and reducing error rates in the results.

Density-Based Optimized K-Means: Density-Based Optimized K-Means was used in this study. In this case, the clusters were assigned to where there is a high density of the data points within a dataset. The clusters are thus assigned wherever there is a high density of data points, separated by low-density areas [23][24]. Most importantly, the user is not required to specify the number of clusters, since there exists a distance-based parameter, which serves as a tunable threshold. The threshold in this respect determines the closeness of the cluster members. The centroids, which represent the center of the clusters, are critical components in the clustering process using Python, in expectation maximization; expectation assigns each data point the nearest centroid. Secondly, the maximization step computes the mean of all the points for every cluster, thereby establishing the new centroids.

The Density-Based Optimized K-Means used in this study differs significantly from the traditional DBSCAN (Density-Based Spatial Clustering of Applications with Noise) algorithm in several key aspects. While both approaches are designed to identify clusters based on data density and do not require the user to predefine the number of clusters, the modified version introduces enhancements that improve scalability and integration with iterative centroid-based refinement. Unlike DBSCAN, which forms clusters through the concept of density reachability and connectivity without relying on centroids, the modified approach incorporates centroid computation and iterative expectation-maximization steps like those used in K-Means. In this process, each data point is assigned to the nearest centroid, and cluster centres are recalculated based on the mean of the assigned points, refining cluster structure with each iteration. Moreover, DBSCAN labels low-density points as noise and does not attempt to reassign them, whereas the modified algorithm performs a secondary scan to reassign previously un-clustered data points and update cluster centres, ensuring more comprehensive coverage. Another critical distinction lies in computational design: the Density-Based Optimized K-Means was specifically structured to support parallelization, making it suitable for large-scale data processing using platforms like Apache Spark. In contrast, DBSCAN's recursive neighbourhood

expansion and dependency on sequential region-growing make it less efficient and harder to scale in parallel computing environments. Thus, while DBSCAN is effective for discovering arbitrary-shaped clusters in smaller datasets, the modified method offers a more structured, centroid-driven, and parallelizable approach that is better suited for big data scenarios and high-performance educational analytics.

The Combiner Process: The Combiner process, also known as the semi-reducer, is an optional step in Spark that summarizes Map output before passing it to the Reducer. In the context of K-Means, it combines results from multiple parallel executions to improve clustering efficiency. The process involves four main steps: (a) the dataset is split into subsets, and K-Means is run on each in parallel; (b) centroids from each subset are collected; (c) these centroids are averaged to form unified cluster centers; and (d) K-Means is re-run on the full dataset using the combined centroids to enhance accuracy. This process is implemented using Spark MLlib.

Cluster Merging: Cluster merging is an unsupervised learning technique that combines smaller clusters with larger ones based on similarity measures. This process simplifies data structure and improves clustering accuracy by refining group boundaries and enhancing overall data representation.

2.3.3.2 Parallelization Implementation of Hierarchical KNN Algorithm

The core of the student behavior prediction model involves using the K-nearest neighbor (KNN) algorithm to predict a student's behavior by analyzing the behavior of their K most similar peers [22]. This process is parallelized in Apache Spark using Resilient Distributed Datasets (RDDs) and involves six main steps. First, training and test data are loaded as RDDs or DataFrames and distributed across worker nodes, with training data optionally broadcasted. Second, the Euclidean distance between test and training points is computed in parallel using Spark's map function. Third, each test point's K nearest neighbors is selected using takeOrdered(), also in parallel. Fourth, predictions are made via majority voting from the K neighbors. Fifth, the predicted results are evaluated by comparing with actual labels, and accuracy is computed in parallel. Finally, performance is optimized by tuning Spark configurations and testing with various dataset sizes to ensure scalability and efficiency.

3 Results and Discussion

3.1 Accuracy Performance of the Hierarchical KNN in Predicting Student Behavior Characteristics

Figure 3 demonstrates that Hierarchical KNN consistently outperforms standard KNN (Without AHP-weighted features filtering) in predicting student behavior across all population sizes, with notably lower average relative error percentages. As the number of students increases from 100 to 10,000, the relative error for KNN fluctuates more significantly, peaking at 9.4% for 5,000 students, while Hierarchical KNN maintains greater stability and lower error rates, ranging from 5.16% to 6.08%. This indicates that Hierarchical KNN scales are more effective and are more robust in handling larger datasets, likely due to its structured clustering approach that reduces computational overhead and improves local neighbor accuracy. The most optimal performance for Hierarchical KNN appears at a population size of 2,000, where it achieves the lowest error of 5.16%, suggesting a potential balance between accuracy and complexity. These findings underscore the practical advantage of Hierarchical KNN for educational institutions, particularly those managing large student populations, as it ensures higher predictive accuracy and greater reliability in real-world deployment scenarios.

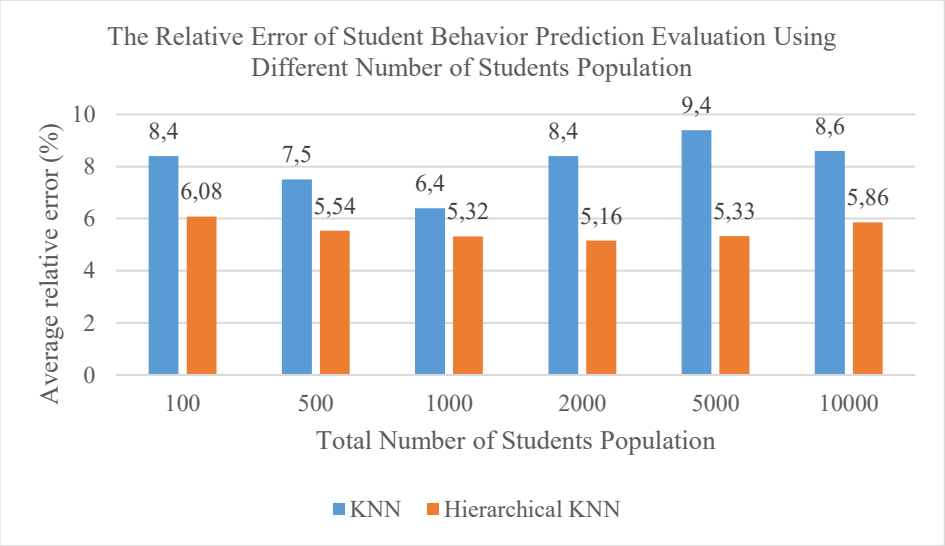

Figure 3 Comparison of Accuracy Performance of the Hierarchical KNN Prediction Vs KNN prediction models using different sizes of student's population.

3.2 Parallelization Approach Performance of the Density-based Optimized K-means

Figure 4 compares the efficiency (running time in seconds) between Spark cluster and standalone mode when clustering different numbers of student data points (ranging from 100 to 10,000). The findings clearly show that the Spark cluster mode offers significantly better scalability and consistent performance across all data sizes. While the running time for Spark cluster remains nearly constant, ranging only slightly from 13.12 seconds (100 students) to 14.36 seconds (10,000 students), the standalone mode becomes increasingly inefficient as the data size grows. Specifically, the standalone running time escalates from 13.12 seconds at 100 students to 300 seconds at 5,000 students, indicating a steep rise in processing cost. This demonstrates that Spark cluster is highly efficient and scalable, handling larger datasets without a significant increase in computation time. In contrast, the standalone setup struggles with larger data volumes, resulting in an over 20x increase in processing time from 100 to 5,000 students. Therefore, for large-scale clustering tasks, Spark cluster is the clearly superior choice, offering both speed and stability.

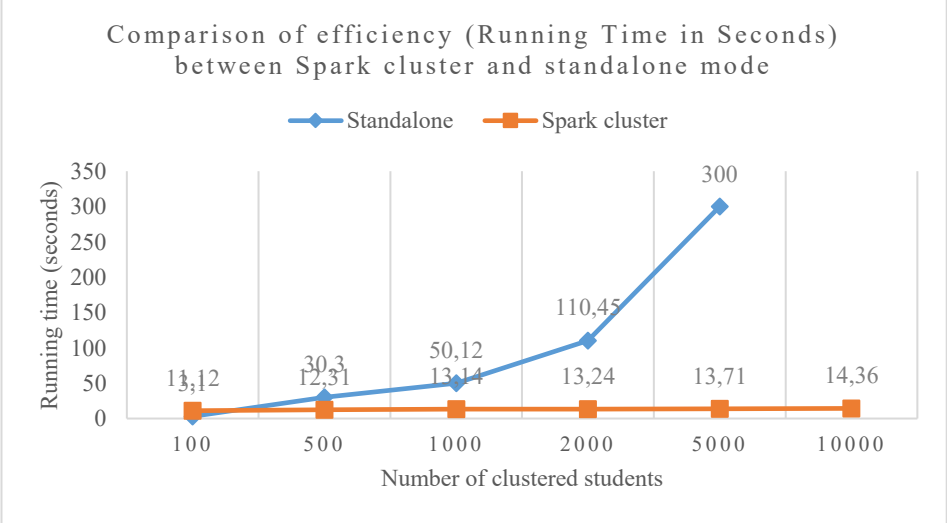

Figure 4 Comparison of efficiency (Running Time in Seconds) between Spark cluster and standalone mode when running Density-based Optimized K-Means clustering process.

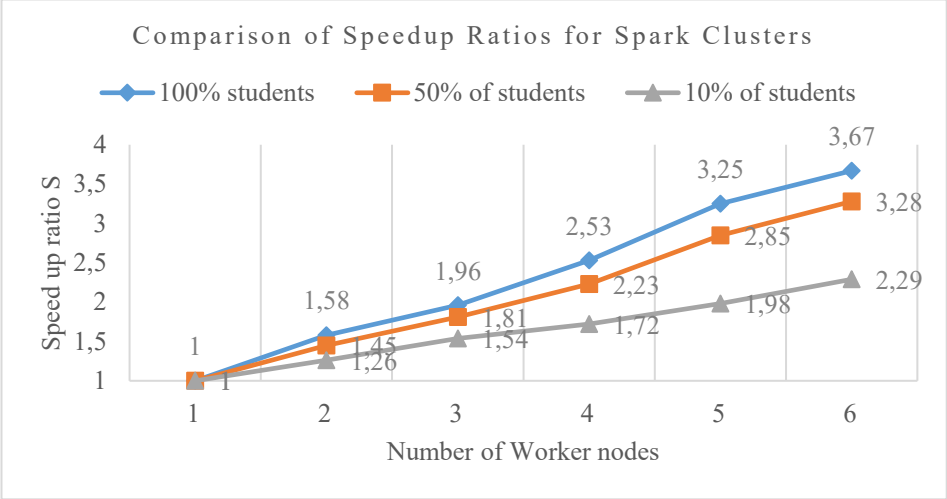

Figure 5 Comparison of Speedup Ratios for Spark Clusters when running Density-based Optimized K-Means clustering process with different number of worker nodes.

The graph shown in Figure 5 presents a comparison of speedup ratios (S) for Spark clusters using different numbers of worker nodes (1 to 6) and varying data volumes, 100%, 50%, and 10% of student datasets. The findings reveal a clear trend: as the number of worker nodes increases, the speedup ratio improves across all dataset sizes, confirming the scalability of Spark for parallel processing. Notably, the largest dataset (100% of students) achieves the highest speedup, reaching 3.67 speedup at 6 nodes, followed by 50% at 3.28 times, while the smallest dataset (10%) reaches only 2.29 times. This shows that larger datasets benefit more significantly from additional worker nodes, as there is more computational workload to distribute. Furthermore, while all configurations show linear or near-linear scalability initially, diminishing returns become visible beyond 4 - 5 nodes, especially for smaller datasets like 10%, where the overhead of parallelization starts to outweigh the benefits. Overall, the graph demonstrates that Spark performs most efficiently when handling larger datasets with enough worker nodes, making it a robust solution for high-volume educational data processing where performance gains scale with resources.

3.3 Computation Performance of the Hierarchical KNN in Predicting Student Behavior Characteristics

Figure 6 illustrates the execution time comparison of the Hierarchical KNN prediction model between standalone mode and Spark cluster across various student population sizes (100 to 10,000). The findings show that Spark cluster

significantly outperforms standalone mode, especially as dataset size increases. For small datasets (e.g., 100 students), both modes show comparable execution times (Spark: 3.15s, Standalone: 3.12s). However, as the dataset grows, the gap widens drastically, at 10,000 students, the standalone mode takes 313 seconds, while the Spark cluster completes in just 39.27 seconds, reflecting an efficiency gain of nearly 8 times. This consistent performance in Spark cluster is attributed to its parallel processing architecture, which effectively distributes the workload and minimizes bottlenecks, even as data volume increases. In contrast, the standalone mode experiences a linear-to-exponential increase in computation time, revealing its limitations for large-scale datasets. Overall, these results highlight that Spark cluster not only accelerates processing for the Hierarchical KNN model but also ensures scalability, efficiency, and practical viability for real-world educational analytics involving large student populations.

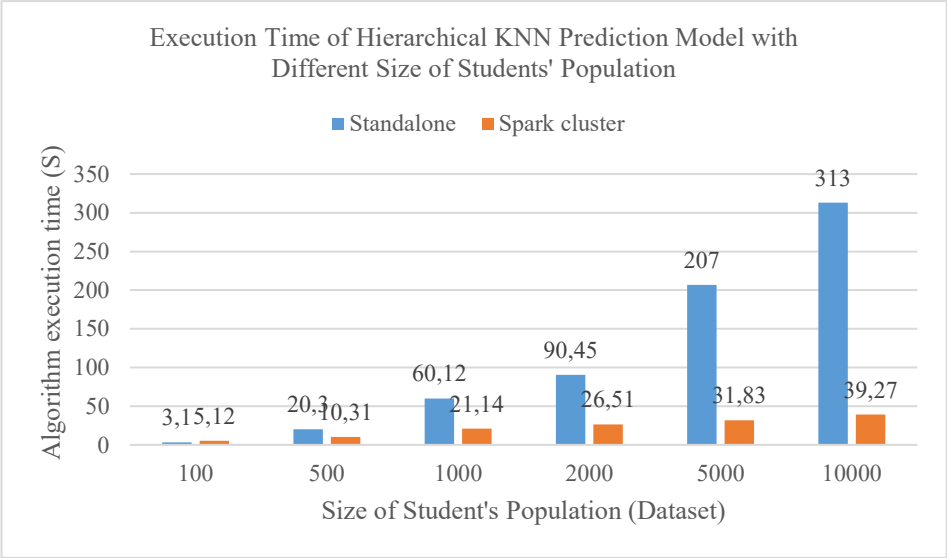

Figure 6 Comparison of Execution Time of the Hierarchical KNN Prediction Model using different sizes of student's population.

The graph, shown in Figure 7, presents the speedup ratios of Spark clusters when executing the Hierarchical KNN algorithm using different numbers of worker nodes (1 to 6) and varying data volumes (100%, 50%, and 10% of students). The results clearly show that speedups improve as more worker nodes are added, with the greatest performance gains observed in the 100% student dataset, which reaches a speedup of 4.97× at 6 nodes. The 50% dataset also scales effectively, achieving 3.88× speedup, while the 10% dataset shows more limited gains, peaking at 2.42×.

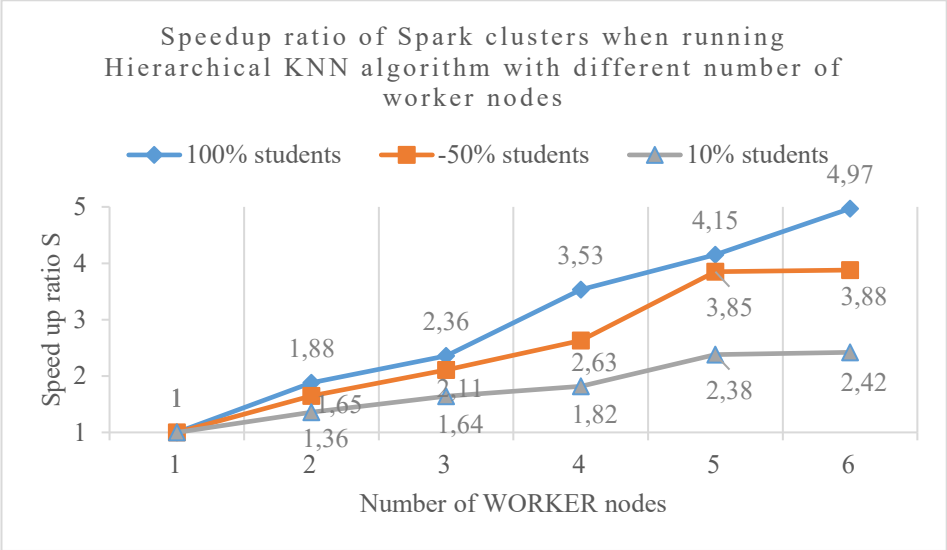

Figure 7 A comparison chart of the speedup ratio of Spark clusters when running Hierarchical KNN algorithm with different number of worker nodes

This indicates that larger datasets benefit more significantly from parallelization, as there is more computation to distribute across nodes. Meanwhile, the smaller dataset experiences diminishing returns beyond 3–4 worker nodes, where parallel overhead may offset performance gains. Overall, the findings confirm that the Spark cluster environment scales efficiently with increasing data volume and worker nodes, making it a highly effective approach for accelerating Hierarchical KNN execution in large-scale educational analytics.

The Density-Based Optimized K-Means algorithm demonstrates effective segmentation of student behavior, and its parallel performance is significantly improved when executed on a Spark cluster compared to a single machine. For large datasets (over 10,000 students), the Spark cluster achieves better scalability, with only slight increases in execution time, whereas the single-machine approach becomes inefficient and overloaded. The speedup ratio improves as the number of worker nodes increases on the Spark cluster, though it doesn't achieve a perfect 1:1 ratio due to data dependencies and communication overhead between nodes. For small datasets, parallelization is less efficient on the Spark cluster due to the overhead of task submission and resource scheduling.

The Hierarchical KNN prediction model shows significant improvements in execution time when implemented in parallel on a Spark cluster, especially when predicting large datasets. The model's performance deteriorates when executed

serially on a single machine, particularly for large datasets, where memory and CPU overload become problematic. The speedup ratio of the Hierarchical KNN model improves as more worker nodes are added to the Spark cluster, demonstrating successful parallelization. However, like K-Means, perfect parallelism is not achieved due to data dependencies and inter-node communication. For small datasets, the overhead of task submission and communication between nodes makes it harder to observe the advantages of parallelization, suggesting that Spark is better suited for handling large-scale datasets.

These findings highlight the scalability and efficiency benefits of implementing both the Density-Based Optimized K-Means and Hierarchical KNN algorithms in parallel on Spark clusters, especially when dealing with large volumes of student data. However, for smaller datasets, the overhead is involved in parallel processing reduces the efficiency gains, indicating that the benefit of parallelization becomes more apparent as dataset size increases. While the parallelization of these models on Spark significantly improves computational efficiency, especially for large datasets. These findings highlight the potential of leveraging big data analytics and machine learning techniques to improve student management and tailor educational interventions. However, parallelization is less effective for smaller datasets, indicating that its benefits scale with larger data volumes. Overall, the study offers a robust framework for analyzing student behaviors, with the potential for widespread application in academic institutions.

The parallelization of the algorithms led to faster execution times, allowing the model to handle large datasets in real-time or near-real-time scenarios. The hybrid model is expected to yield more accurate predictions than any individual method, contributing to better decision-making for interventions based on student behavior predictions.

4 Limitations of the Proprosed Solution

Despite its strong performance, the model has potential limitations. Its reliance on AHP for feature weighting introduces subjectivity, as the assignment of feature importance depends on expert judgment and may lead to bias if not rigorously validated. Furthermore, while the use of Apache Spark enables high scalability, deploying such a system in real-world educational environments may be challenging due to limited computational resources, infrastructure constraints, and technical expertise. In resource-constrained institutions, the cost and complexity of maintaining distributed computing environments may limit the model's applicability and necessitate simplified or lightweight alternatives. Nevertheless, the findings highlight the model's value as a robust, high-performance framework for real-time behaviour analysis, academic risk

detection, and data-driven educational intervention, particularly in data-intensive and well-resourced environments.

To enhance the robustness, accessibility, and applicability of the proposed hybrid model across diverse educational environments, several improvements are recommended. To reduce the subjectivity in AHP-based feature weighting, future studies should incorporate recent data-driven techniques such as SHAP [26], permutation feature importance, and ensemble-based ranking systems to validate and complement expert decisions. These methods enhance objectivity and support explainability in machine learning applications. In low-resource educational institutions, lightweight model deployment using multiprocessing within scikit-learn, Dask, or on-demand environments like Google Colab and AWS SageMaker Studio Lab [30] offers an affordable and scalable alternative. For flexible deployment, recent advancements in container orchestration using Docker and Kubernetes have made modular, reproducible deployment of AI systems more feasible [27]. Furthermore, adaptive model scaling, where computational complexity adjusts to available resources, can be implemented with AutoML-based configurations optimized for edge computing environments [29], allowing decentralized processing with minimal latency and infrastructure dependence. Crucially, data summarization techniques should be integrated into the data preprocessing pipeline to reduce redundancy, enhance model efficiency, and enable real-time analytics, particularly when handling high-dimensional student behavior data [31-33]. Cross-validation across international and resource-diverse academic settings is vital to establish the model's generalizability; studies such as those by Ortega et al. (2024) stress the need for inclusive validation strategies in AI for education [28]. Lastly, developing intuitive user interfaces and automation pipelines can lower the barrier for non-technical users. Modern visual analytics tools like Streamlit and no-code platforms [25] enable educators to explore model outputs, run predictions, and adjust parameters without coding, making AI-powered educational analytics more accessible. These directions ensure that the proposed hybrid model becomes not only technically sound but also widely usable, scalable, and interpretable across varied educational ecosystems.

It should also be noted that the dataset analyzed in this study spans from 2015 to 2017, a period prior to the global shift toward large-scale online and blended learning environments. While the model's methodological contributions in scalability, feature weighting, and parallelized prediction remain valid, the generalizability of the findings should be further tested on more contemporary datasets. Post-2020, particularly during and after the COVID-19 pandemic, student learning behaviors have undergone significant transformations due to increased reliance on digital platforms, remote learning, and online assessments. These changes may introduce new behavioral features (e.g., login frequency,

virtual classroom participation, online discussion forum activity) that were less prevalent in earlier datasets. Future work should therefore evaluate the proposed model using post-2020 student data to confirm its robustness and adaptability in modern digital learning ecosystems.

5 Conclusion

This study addresses the growing need for sophisticated data mining techniques to predict student behaviour by designing and evaluating a parallelized hybrid model that integrates Density-Based Optimized K-Means clustering, Analytic Hierarchy Process (AHP) for feature weighting, and K-Nearest Neighbours (KNN) for student behaviour prediction. The research successfully achieves its objectives, offering a scalable and efficient framework to predict student behaviour based on real-time data analytics. Based on the findings, parallelization on the Spark cluster enhances computational efficiency for both Density-Based K-Means and Hierarchical KNN, especially when handling large datasets (over 10,000 students). The execution times of the algorithms on Spark increase minimally with dataset size compared to single-machine implementations, which become overwhelmed by larger data volumes. The parallelized hybrid approach of combining clustering, feature weighting, and KNN demonstrates superior execution efficiency, making it a valuable tool for educational institutions to monitor and intervene in student progress in real-time. However, parallelization proves less beneficial for smaller datasets due to the overhead involved, indicating the model's true value emerges in large-scale applications.

While the current study is based on historical student data, the proposed Hierarchical KNN prediction model has strong potential for real-time deployment within Learning Management Systems (LMS) such as Moodle, Canvas, or Blackboard. By integrating the model into the LMS backend, real-time student activity data, such as login frequency, assignment submissions, forum participation, and quiz performance, can be continuously collected and analyzed. This enables the model to dynamically monitor student behavior, identify at-risk learners early, and trigger timely interventions such as automated alerts, personalized feedback, or academic support recommendations. Using platforms like Apache Spark for distributed processing, the system can handle large-scale, real-time data streams efficiently, even in institutions with thousands of active users. Additionally, containerization tools (e.g., Docker) and API-based deployment can facilitate seamless integration with existing LMS infrastructure, ensuring that the solution is both scalable and maintainable in a production environment. Ultimately, embedding this model into real-time educational ecosystems enhances its practical impact, promoting proactive learning support and data-driven decision-making in digital education.

Acknowledgement

The authors would like to express their sincere gratitude to Shandong Light Industry Vocational College and Universiti Malaysia Sabah for their financial support, which made this research possible.

References

[1] Luo, Y., Han, X. & Zhang, C., *Prediction of Learning Outcomes with a Machine Learning Algorithm based on Online Learning Behavior Data in Blended Courses,* Asia Pacific Education Review, **25**(2), 267-285, 2024. DOI: 10.1007/s12564-022-09749-6

[2] Alhammadi, A., Shayea, I., El-Saleh, A.A., Azmi, M.H., Ismail, Z.H., Kouhalvandi, L. & Saad, S. A., *Artificial Intelligence in 6G Wireless Networks: Opportunities, Applications, and Challenges,* International Journal of Intelligent Systems, **2024**, 8845070, 2024. DOI: 10.1155/2024/8845070

[3] Ng, T.K., *New Interpretation of Extracurricular Activities Via Social Networking Sites: A Case Study of Artificial Intelligence Learning at a Secondary School in Hong Kong*, Journal of Education and Training Studies, **9**(1), pp.49-60, 2021. DOI: 10.11114/jets.v9i1.5105

[4] Camerer, C.F., *Artificial Intelligence and Behavioral Economics*, in Agrawal, A., Gans, J. & Goldfarb, A. (eds), *The Economics of Artificial Intelligence: An Agenda*, 587-608, University of Chicago Press, 2019. DOI: 10.7208/chicago/9780226613475.001.0001

[5] Hu, J., Huang, Z., Li, J., Xu, L. & Zou, Y., *Real-time Classroom Behavior Analysis for Enhanced Engineering Education: An AI-assisted Approach,* International Journal of Computational Intelligence Systems, **17**(1), 167, 2024. DOI: 10.1007/s44196-024-00572-y

[6] Yağcı, M., *Educational Data Mining: Prediction of Students' Academic Performance using Machine Learning Algorithms*, Smart Learn. Environ, **9**, 11, 2022. DOI: 10.1186/s40561-022-00192-z

[7] Song, X., *Student Performance Prediction Employing K-nearest Neighbor Classification Model and Meta-heuristic Algorithms. Multiscaleand Multidisciplinary Modeling,* Experiments and Design, 1-16, 2024. https://doi.org/10.1007/s41939-024-00481-9

[8] Alqatow, I., Rattrout, A. & Jayousi, R., *Prediction of Student Performance with Machine Learning Algorithms based on Ensemble Learning Methods.* In: Zhang, F., Wang, H., Barhamgi, M., Chen, L., Zhou, R. (eds) Web Information Systems Engineering – WISE 2023. WISE 2023. Lecture Notes in Computer Science, **14306**, Springer, Singapore, 2023. DOI: 10.1007/978-981-99-7254-8_40

[9] Chen, Y. & Zhai, L., *A Comparative Study on Student Performance Prediction using Machine Learning,* Educ. Inf. Technol., **28**, pp. 12039-12057, 2023. DOI: 10.1007/s10639-023-11672-1

[10] Yang, S., Choi, J., Bae, S. & Chung, M., *A Hybrid Prediction Model Integrating FCM Clustering Algorithm with Supervised Learning.* In: Park, DS., Chao, HC., Jeong, YS., Park, J. (eds) Advances in Computer Science and Ubiquitous Computing. Lecture Notes in Electrical Engineering, **373**. Springer, Singapore, 2015. https://doi.org/10.1007/978-981-10-0281-6_88

[11] Hajirahimi, Z. & Khashei, M., *A Novel Parallel Hybrid Model based on Series Hybrid Models of ARIMA and ANN Models,* Neural Process Lett, **54**, pp. 2319-2337, 2022. DOI: 10.1007/s11063-021-10732-2

[12] Khotimah, B.K., Anamisa, D.R., Kustiyahningsih, Y., Fauziah, A.N. & Setiawan, E., *Enhancing Small and Medium Enterprises: A Hybrid Clustering and AHP-TOPSIS Decision Support Framework.* Ingénierie des Systèmes d'Information, **29**(1), pp. 313-321, 2024. DOI: 10.18280/isi.290131

[13] Al-Sayed, Amna, Mashael, M., Khayyat, & Nuha Zamzami, *Predicting Heart Disease using Collaborative Clustering and Ensemble Learning Techniques.* Applied Sciences **13**(24), 13278, 2023. DOI: 10.3390/app132413278

[14] Maddukuri, C.D. & Senapati, R., *Hybrid Clustering-based Fast Support Vector Machine Model for Heart Disease Prediction,* In: Udgata, S.K., Sethi, S., Gao, XZ. (eds) Intelligent System, ICMIB 2023. Lecture Notes in Networks and Systems, **728**, Springer, Singapore, 2024. DOI: 10.1007/978-981-99-3932-9_24

[15] Zhang, L., Zhu, Y., Su, J., Lu, W., Li, J. & Yao, Y., *A Hybrid Prediction Model based on KNN-LSTM for Vessel Trajectory,* Mathematics, **10**(23), 4493, 2022. DOI: 10.3390/math10234493

[16] Dziewior, J., Carr, L.J., Pierce, G.L. & Whitaker, K., *College Students Report Less Physical Activity and More Sedentary Behavior during the COVID-19 Pandemic,* Journal of American College Health, **72**(7), pp. 2022-2030, 2024. DOI: 10.1080/07448481.2022.2100708

[17] Shen, X. & Yuan, C., *A College Student Behavior Analysis and Management Method Based on Machine Learning Technology,* Wireless Communications and Mobile Computing, **2021**, pp. 1-10, 2021. DOI: 10.1007/978-3-030- 89508-2_19

[18] Li, X., Zhang, Y., Cheng, H., Zhou, F. & Yin, B., *An Unsupervised Ensemble Clustering Approach for the Analysis of Student Behavioral Patterns,* IEEE Access, **9**, pp. 7076-7091, 2021. DOI: 10.1109/ACCESS.2021.3049157

[19] Ding, D., Li, J., Wang, H. & Liang, Z., *December. Student Behavior Clustering Method based on Campus Big Data,* in 2017 13th International

Conference on Computational Intelligence and Security (CIS), pp. 500-503, IEEE, 2017. DOI: 10.1109/CIS.2017.00116

[20] Ali El-Sayed Ali, H., Alham, M.H. & Ibrahim, D.K., *Big Data Resolving using Apache Spark for Load Forecasting And Demand Response in Smart Grid: A Case Study of Low Carbon London Project.* Journal of Big Data, **11**(1), 59, 2024. DOI: 10.1186/s40537-024-00909-6

[21] Pourahmad, S., Basirat, A., Rahimi, A. & Doostfatemeh, M., *Does the Determination of Initial Cluster Centroids Improve the Performance of the Clustering Algorithm? Comparison of Three Hybrid Methods by Genetic Algorithm, Minimum Spanning Tree, and Hierarchical Clustering* in *An Applied Study,* Computational and Mathematical Methods in Medicine, 2020. DOI: 10.1155/2020/7636857

[22] Ahmed, M.A., Baharin, H. & Nohuddin, P.N., *Analysis of K-means, DBSCAN, and OPTICS Cluster Algorithms on Al-quran Verses,* International Journal of Advanced Computer Science and Applications, **11**(8), 248-254, 2020. DOI: 10.14569/IJACSA.2020.0110832

[23] Yang, K., Mohammadi Amiri, M. & Kulkarni, S.R., *Greedy Centroid Initialization for Federated K-means.* Knowledge and Information Systems, 1-33, 2024. DOI: 10.1109/CISS56502.2023.10089666

[24] Fränti, P. & Sieranoja, S., *How Much Can K-means be Improved by using Better Initialization and Repeats?,* Pattern Recognition, **93**, 95-112, 2019. DOI: 10.1016/j.patcog.2019.04.014

[25] Truss, M. & Schmitt, M., *Human-centered AI Product Prototyping with No-code Automl: Conceptual Framework, Potentials and Limitations,* International Journal of Human–Computer Interaction, 1-16, 2024. DOI: 10.1080/10447318.2024.2425454

[26] Hasan, A.S., Jalayer, M., Das, S. & Kabir, M.A.B., *Application of Machine Learning Models and SHAP to Examine Crashes Involving Young Drivers in New Jersey,* International Journal of Transportation Science and Technology, **14**, pp. 156-170, 2024. DOI:.1016/j.ijtst.2023.04.005

[27] Pamadi, E.V.N., Khan, S. & Goel, E,O., *A Comparative Study on Enhancing Container Management with Kubernetes,* International Journal of Advanced Research and Interdisciplinary Scientific Endeavours, **1**(3), pp. 116-133, 2024. DOI: 10.61359/11.2206-2411

[28] Edeni, C.A., Adeleye, O.O. & Adeniyi, I.S., *The Role of AI-enhanced Tools in Overcoming Socioeconomic Barriers in Education: A Conceptual Analysis,* World Journal of Advanced Research and Reviews, **21**(3), pp. 944-951, 2024. DOI: 10.30574/wjarr.2024.21.3.0780

[29] Wang, Y., Yang, C., Lan, S., Zhu, L., & Zhang, Y., *End-edge-cloud Collaborative Computing for Deep Learning: A Comprehensive Survey.* IEEE Communications Surveys & Tutorials, **26**(4), pp. 2647-2683, 2024. DOI: 10.1109/COMST.2024.3393230

[30] Cherukuri, B.R., *Serverless Computing: How to Build and Deploy Applications without Managing Infrastructure,* World Journal of Advanced Engineering Technology and Sciences, **11**(2), pp. 650-663, 2024. DOI: 10.30574/wjaets.2024.11.2.0074

[31] Alfred, R., *Summarizing Relational Data using Semi-supervised Genetic Algorithm-based Clustering Techniques,* Journal of Computer Science, **6**(7), 775, 2010.

[32] Alfred, R. & Kazakov, D., *Data Summarization Approach to Relational Domain Learning based on Frequent Pattern to Support the Development of Decision Making,* In International Conference on Advanced Data Mining and Applications, (pp. 889-898), Berlin, Heidelberg: Springer Berlin Heidelberg, August, 2006. DOI: 10.1007/11811305_97

[33] Alfred, R., *DARA: Data Summarisation with Feature Construction.* in 2008 Second Asia International Conference on Modelling & Simulation (AMS) (pp. 830-835), IEEE, May, 2008. DOI: 10.1109/AMS.2008.131

[34] Sainin, M.S., Alfred, R. & Ahmad, F., *Ensemble Meta Classifier with Sampling and Feature Selection for Data with Imbalance Multiclass Problem,* Journal of Information and Communication Technology, **20**(2), pp. 103-133. 2021. DOI: 10.32890/jict2021.20.2.1.