IJMSHE
International Journal of Multidisciplinary Studies in Higher Education

International Journal of Multidisciplinary Studies in Higher Education

Journal Homepage: https://ijmshe.com/index.php/pub/index

Research Article

Leveraging Prolog's Declarative Power for Clustering Student Performance in a Timed Quiz

Malakit L. Ram [1] | Renee Clint P. Gortifacion [2] | Jose C. Agoylo Jr [3] | Jorton A. Tagud [4]

[1-4] Southern Leyte State University

Article Info

Article history:

Received: 03 January 2025
Revised: 14 January 2025
Accepted: 8 February 2025

Keywords:

Prolog programming, K-Means clustering, Student performance analysis.

Abstract

Student performance analysis in programming education presents challenges in identifying learning patterns and addressing diverse needs. This study presents an automated Prolog-based quiz program that assesses student learning progression using a structured quiz system. In contrast to most assessment tools, the Prolog program uses its declarative nature to dynamically generate questions, give feedback in real time, and allow level-based progression. This makes sure that the learning experience is both interactive and flexible. Data collected from 140 students during a 15-minute quiz session was analyzed using the K-Means clustering algorithm, which grouped students into three clusters: foundational learners, intermediate learners, and advanced performers. Silhouette scores validated the robustness of the clustering results, demonstrating well-defined groupings. Visualizations, including scatter plots, bar charts, and box plots, highlighted distinct performance profiles, showing that most students struggled with intermediate levels, while fewer mastered advanced topics. These results show how important it is to tailor educational interventions and how useful it is to use both Prolog-based tests and clustering methods for large-scale educational data mining. Beyond programming education, this framework has potential applications in other domains, such as logic-driven problem-solving courses, computational thinking modules, and adaptive learning environments. Integration of Prolog-based tests into blended learning systems, the use of different clustering algorithms, and the addition of engagement metrics to get a fuller picture of student performance should all be looked into in future research. By addressing these areas, this study aims to contribute to the advancement of adaptive educational tools and personalized learning strategies.

Cite as: Ram, M., Gortifacion, R. C., Agoylo Jr., J., & Tagud, J. (2025). Leveraging Prolog's Declarative Power for Clustering Student Performance in a Timed Quiz. International Journal of Multidisciplinary Studies in Higher Education, 2(1). https://doi.org/10.70847/592808

[1]Corresponding Author: Malakit L. Ram
[*]Corresponding Email: malakitr21@gmail.com

Introduction

K-means clustering and its variations have been studied a lot in many fields, such as software development (Almansoury et al., 2022) and analyzing student performance (Veneta Tabakova-Komsalova et al., 2023). The K-Means algorithm's simplicity and efficiency have made it a widely used clustering technique (Ahmed et al., 2020). In educational contexts, it has proven effective for analyzing student performance patterns and informing targeted interventions. New improvements, like hierarchical K-Means (Sinaga & Yang, 2020) and robust deep clustering models (Huang et al., 2021), have made it even better at handling big datasets, finding complicated patterns, and giving useful information. These changes show how important clustering techniques are becoming for dealing with the problems that come up when students learn in different ways.

Prolog, as a declarative programming language, offers significant advantages over imperative languages in educational settings. Unlike imperative languages, which require detailed step-by-step instructions, Prolog allows students to focus on defining relationships and logical rules to solve problems. This shift from procedural thinking to declarative reasoning is particularly beneficial for developing computational thinking skills, as it encourages learners to approach problems from a logical and systematic perspective. The declarative nature of Prolog also supports a higher degree of abstraction, enabling students to focus on problem-solving strategies rather than the mechanics of code implementation. Prolog's emphasis on logic and inference aligns well with the needs of educational contexts that aim to develop higher-order thinking skills, making it a powerful tool for teaching problem-solving and critical reasoning.

Clustering methods, like K-Means, are very important in adaptive learning systems because they make it possible to find performance patterns and learner profiles. Educators can personalize instructional strategies with these insights, tailoring interventions to meet the specific needs of diverse learners. For instance, clustering can reveal gaps in foundational knowledge, highlight transitional challenges in intermediate learners, and identify advanced learners who are ready for more complex tasks. Beyond individual support, clustering offers institutional benefits by providing actionable data that can inform curriculum design, optimize resource allocation, and improve overall instructional quality. Clustering has a bigger effect in adaptive learning because it can make personalized education more widespread and help make decisions about curriculum design and how to teach based on data.

Even though clustering and Prolog could be useful in education, more research is needed on how to combine them to solve domain-specific problems, like teaching Prolog programming. While Prolog's declarative approach offers unique advantages, many existing educational frameworks focus on imperative languages like Python or Java, leaving limited exploration of its pedagogical potential. Similarly, previous studies have largely focused on clustering applications in general educational contexts (Veneta Tabakova-Komsalova et al., 2023; Govender & Sivakumar, 2020) or adaptive learning environments, with limited exploration of their role in fostering computational skills through dynamic, real-time assessments. This gap is particularly evident in assessments that require adaptability and responsiveness, which are critical for personalized learning. Also, current assessment methods rely on static tools that don't give learners the adaptive feedback they need for personalized learning. This leaves a major gap in providing effective instructional support for students.

This study addresses these gaps by introducing a Prolog-based quiz program that leverages the language's declarative nature to create an adaptive, interactive learning assessment tool.

[1]Corresponding Author: Malakit L. Ram
[*]Corresponding Email: malakitr21@gmail.com

By dynamically generating questions, providing immediate feedback, and facilitating level-based progression, the program offers a novel method for evaluating student performance in Prolog programming. K-Means clustering is employed to analyze the performance data generated by the quiz, identifying distinct clusters of learners and their implications for curriculum design. Combining Prolog's unique strengths with clustering analysis makes this research a step forward in the field of educational technology. It provides a flexible and scalable framework for judging student performance. These additions are meant to improve the way programming is used in schools now and make it easier for adaptive tests and data-driven teaching methods to be used in more places. This framework could lead to the creation of similar tools for other declarative programming languages besides Prolog. This would have a bigger effect on computational education and adaptive learning methods (Ahmed et al., 2020; Sinaga & Yang, 2020; Huang et al., 2021).

Theoretical Framework

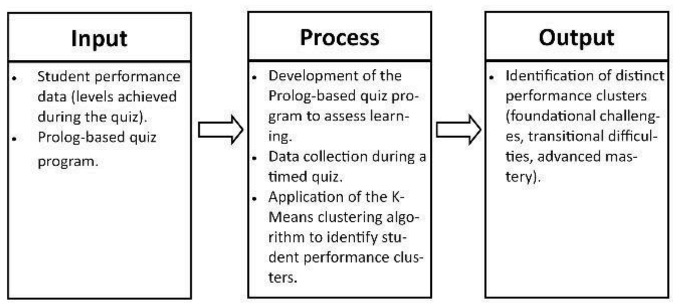

Figure 1. Input Process Output framework

Figure 1 adopted in this study serves as a structured theoretical foundation for analyzing the flow of activities from data collection to result interpretation. This model highlights the systematic transformation of inputs into meaningful outputs through defined processes, aligning seamlessly with the study's objectives.

The input phase focuses on gathering student performance data derived from a Prolog-based quiz. This dataset includes levels achieved during a timed 15-minute session, forming the foundational dataset for subsequent analysis. By capturing critical performance indicators, the input phase sets the groundwork for understanding individual and group learning trajectories.

The process phase operationalizes the transformation of raw data into actionable insights. It comprises three key activities:

1. Development of the Prolog-Based Quiz Program: The quiz program leverages Prolog's declarative nature, allowing for dynamic question generation, real-time feedback, and structured progression through five levels of increasing difficulty. This step emphasizes adaptive learning and individualized assessment.

[1]Corresponding Author: Malakit L. Ram
*Corresponding Email: malakitr21@gmail.com

2. Data Collection: During the quiz, student interactions are recorded, providing a rich dataset of learning patterns and progression.

3. Application of the K-Means Clustering Algorithm: The clustering algorithm groups students into performance clusters, offering insights into foundational challenges, transitional difficulties, and advanced mastery.
The output phase synthesizes the processed data into actionable results. The identification of distinct clusters, students are grouped into performance categories, revealing varied levels of engagement and proficiency. Insights for tailored instructional strategies, these clusters provide the basis for designing targeted educational interventions, such as reinforcement of foundational concepts, scaffolding for intermediate learners, and advanced problem-solving tasks for high achievers.

The IPO framework aligns with constructivist learning theory, which emphasizes active knowledge construction through meaningful experiences. By facilitating dynamic question generation and real-time feedback, the Prolog-based quiz fosters an environment where learners engage actively with the content, reflecting the principles of constructivism. Moreover, the clustering process highlights areas where scaffolding and differentiated instruction—both central to constructivist approaches—can be effectively applied. For instance, students in lower-performing clusters may benefit from foundational reinforcement, while advanced learners can be challenged with exploratory problem-solving tasks.

From a theoretical standpoint, clustering techniques such as K-Means serve as vital tools in educational data mining (EDM), which seeks to uncover hidden patterns in educational datasets. The grouping of students into distinct performance clusters provides a foundation for differentiated learning, enabling personalized educational experiences tailored to individual needs. The theoretical underpinnings of clustering emphasize its ability to capture variability in learning trajectories, identify performance gaps, and inform adaptive learning systems. By translating raw data into meaningful educational insights, clustering contributes to the broader goals of personalized learning and data-driven curriculum design.

The IPO framework serves as an effective bridge between theoretical concepts and their practical implementation in educational research. By connecting structured data flow with performance analysis, this framework ensures a logical progression from raw data collection to actionable insights. It integrates principles of constructivism, differentiated learning, and educational data mining, underscoring its relevance and applicability in advancing adaptive learning systems and personalized instruction.

Materials and Methods

Program Development

The primary objective of this study was to develop an automated Prolog-based quiz program designed to assess and enhance student learning through a structured quiz system. The program dynamically generates and evaluates questions across five levels of increasing difficulty, ensuring that students must achieve a minimum score to progress to the next level.

[1]Corresponding Author: Malakit L. Ram
*Corresponding Email: malakitr21@gmail.com

The choice of five levels was based on the progressive learning stages inherent in Prolog programming. These levels correspond to a gradual transition from foundational concepts, such as syntax and simple operations, to advanced topics like recursion and meta-predicates. This structure reflects a scaffolded learning approach, enabling students to build competence step-by-step.

Program Structure

The program uses a Prolog knowledge base, which consists of five levels of questions, each level containing 10 questions that progressively increase in complexity. Topics range from basic Prolog syntax and operations to more advanced concepts, providing a comprehensive assessment of a student's understanding of logic programming. question format, each question includes a unique identifier, a question statement, and a correct answer. Here are some of the codes:

- question(1, 1, 'What is 2 + 2 in Prolog?', 4).
- question(2, 4, 'What does "member(X, [1, 2, 3])." return for X?', '1').

Core Functionality

The program consists of the following key components:

- Dynamic Question Selection, questions are randomly selected for each level using a custom random subset generator.
- Level-Based Progression, students must score at least 7/10 (70%) to proceed to the next level. If they fail, the program restarts the same level until they pass.
- Immediate Feedback, each response is evaluated in real-time, providing instant feedback on correctness, along with the correct answer if the response is wrong.
- Completion Check, the program concludes once a student successfully passes all five levels.

Implementation

The system was implemented using Prolog, leveraging its declarative nature and built-in support for logical inference. Key predicates include:

- start_quiz/0: Initializes the quiz and starts from Level 1.
- start_level/1: Manages progression and reattempts for each level.
- ask_questions/4: Handles question presentation, user input, and score calculation.
- random_questions/3: Dynamically selects a subset of 10 questions for each level.

During implementation, challenges such as ensuring random yet balanced question distribution, handling invalid user inputs, and designing robust predicates for real-time feedback were encountered. These issues were

[1]Corresponding Author: Malakit L. Ram
[*]Corresponding Email: malakitr21@gmail.com

addressed by iterative testing and refining Prolog predicates to handle edge cases effectively.

Program Functionality

When the quiz begins, the program welcomes the user and starts from Level 1. Students are presented with one question at a time, where they must type their answer exactly as expected and end with a period. The program evaluates the response and moves to the next question and provides corrective feedback. Upon completing all questions for a level, the program checks the student's score and determines whether they pass or need to retake the level. Once a student completes all levels, the program concludes with a congratulatory message. This program served as the basis for collecting data on student performance, which was subsequently analyzed using the K-Means clustering algorithm to identify learning patterns and guide educational strategies.

Data Collection

The study utilized performance data from 140 students who participated in a Prolog programming quiz. The quiz consisted of five progressive levels, each containing 10 randomized questions. A 15-minute time limit was chosen to simulate a controlled and focused testing environment, emphasizing both speed and accuracy. This constraint aimed to identify performance bottlenecks and provide insights into how students respond under time pressure.

The dataset primarily captured the highest level completed by each student during the quiz. This metric served as a direct indicator of their progression through the quiz and their ability to master increasingly complex Prolog concepts. By focusing on the achieved level, the study was able to simplify the analysis while still providing valuable insights into student performance. This data points to students' proficiency in Prolog programming, their ability to apply logical reasoning, and their progression through foundational to advanced topics.

Although additional metrics such as accuracy or time per question were not included, the captured level alone provided sufficient granularity to group students into distinct performance clusters. For example, students who consistently reached higher levels demonstrated a deeper understanding of Prolog's declarative logic, whereas those who remained at lower levels indicated a need for targeted support in foundational concepts. This focus on levels also aligned with the adaptive nature of the quiz, which progressively introduced more complex questions, making the highest level completed a reliable measure of student performance.

The time-bound nature of the quiz further emphasized the importance of the achieved level, as students needed to balance logical accuracy with time management. The uniformity of the testing environment ensured that external factors, such as distractions, were minimized, providing a reliable context for evaluating the data. By using the highest level as the central metric, the study simplified the dataset while retaining its relevance for clustering and educational insights.

[1]Corresponding Author: Malakit L. Ram
[*]Corresponding Email: malakitr21@gmail.com

Preprocessing

The dataset was preprocessed to ensure accuracy and consistency:

- Missing or incomplete data entries were removed.
- Duplicate entries were identified and eliminated.
- The data was normalized for effective clustering, with categorical values converted into numerical representations where necessary.

Data Mining

The K-Means clustering algorithm was employed to group students based on their performance levels. The decision to use K-Means was influenced by its simplicity, computational efficiency, and suitability for moderate-sized datasets like the one used in this study. Alternative algorithms, such as hierarchical clustering and DBSCAN, were considered but deemed less appropriate due to their complexity and sensitivity to noise in small datasets. K-Means offered an optimal balance of interpretability and performance, making it ideal for grouping students into well-defined clusters. The clustering results were visualized using scatter plots with centroids, cluster size bar charts, and level distribution box plots:

- Scatter Plot with Centroids: Showed the spread of students across clusters.
- Cluster Size Bar Chart: Depicted the distribution of students in each cluster.
- Level Distribution Box Plot: Provided insights into the level range and variability within each cluster.

Software and Tools

The clustering and visualizations were performed using Python programming language and libraries such as:

- scikit-learn: For implementing K-Means clustering.
- Matplotlib and Seaborn: For data visualization.
- Pandas: For data manipulation and analysis.

Evaluation Metrics

The clustering outcomes were evaluated using the following metrics:

The Silhouette score, measures how well students were grouped within clusters. For each data point i, the Silhouette score $S(i)$ is calculated as:

$$S(i) = \frac{b(i)-a(i)}{(a(i), b(i))}$$

Where:

[1]Corresponding Author: Malakit L. Ram
[*]Corresponding Email: malakitr21@gmail.com

- $a(i)$: The average intra-cluster distance (i.e., the average distance between iii and all other points within the same cluster).
- $b(i)$: The average inter-cluster distance (i.e., the average distance between iii and all points in the nearest cluster).

The overall silhouette score is the mean of $S(i)$ for all points:

$$S = \frac{1}{N}\sum_{i=1}^{N} S(i)$$

Where N is the total number of data points. The score ranges from -1 to 1:

- $S > 0$: Indicates well-clustered data.
- $S \approx 0$: Indicates overlapping clusters.
- $S < 0$: Indicates incorrect clustering.

Cluster Cohesion:

Cohesion C_k measures intra-cluster similarity how tightly point within a cluster k are grouped. It is calculated as the sum of squared distances of all points x_i in cluster k from the cluster centroid μ_k:

$$C_k = \sum_{x_i \in k} \| x_i - \mu_k \|$$

Lower C_k values indicate tighter clustering and higher intra-cluster similarity.

Cluster Separation:

Separation $S_{k,j}$ measures inter-cluster differences the distance between the centroids of two clusters k and j:

$$S_{k,j} = \| \mu_k - \mu_j \|$$

Where μ_k and μ_j are the centroids of clusters k and j, respectively. Higher $S_{k,j}$ values indicate better-separated clusters.

Addressing Biases

Although the dataset comprised 140 students, the limited sample size may introduce biases, such as underrepresentation of certain performance groups or variability in prior programming experience. Future studies could address this limitation by incorporating larger

[1]Corresponding Author: Malakit L. Ram
[*]Corresponding Email: malakitr21@gmail.com

and more diverse datasets. Additionally, the timed nature of the quiz could disproportionately affect students who require more time to process questions, highlighting the need to balance time constraints with fair assessment. Furthermore, individual factors such as varying levels of familiarity with Prolog, differences in learning styles, and prior exposure to declarative programming languages may have influenced the results, underscoring the importance of designing assessments that are inclusive and representative of diverse learner profiles.

Ethical Considerations

Participation data was anonymized to maintain student privacy. The study adhered to ethical guidelines for data usage and ensured that no personally identifiable information was included. All participants provided informed consent prior to the study, ensuring voluntary participation and compliance with institutional ethical standards.

Results and Discussion

The findings presented below provide a detailed understanding of student clustering and performance patterns. These results set the stage for discussing their implications for educational practices and policy.

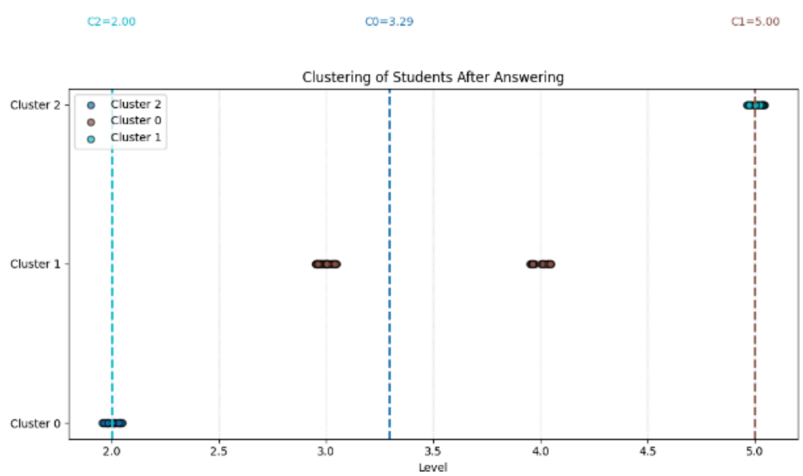

Figure 2. Scatter Plot of Student Clusters with Centroids

The scatter plot (Figure 2) with centroids illustrates how students were clustered based on their achieved levels after 15 minutes of answering the programmed Prolog quiz. The x-axis represents the levels achieved (2, 3, 4, 5), while the y-axis separates the three clusters identified by the K-Means algorithm: Cluster 2 (low achievers), Cluster 0 (mid-level achievers), and Cluster 1 (high achievers). Each cluster is represented by distinct colors, with vertical dashed lines indicating the cluster centroids: Cluster 2 at level 2.0, Cluster 0 at level 3.29, and Cluster 1 at level 5.0.

[1]Corresponding Author: Malakit L. Ram
[*]Corresponding Email: malakitr21@gmail.com

The plot reveals that most students were concentrated in Clusters 2 and 0, signifying lower and intermediate performance levels, respectively, while fewer students progressed to Cluster 1, which represents the highest level achieved. Notably, outlier behavior was observed in Cluster 0, where a few students approached the performance of advanced learners (Cluster 1). These outliers suggest the potential for these students to excel with additional scaffolding and targeted interventions. External factors such as prior experience in programming, familiarity with Prolog syntax, or cognitive adaptability might have influenced their placement. This distribution suggests that the 15-minute time constraint might have limited students' ability to progress further. The clustering outcomes align with differentiated instructional theories, suggesting that tailored support could address the challenges faced by lower-performing groups. Unexpectedly, Cluster 2 students showed little spread, indicating consistent foundational challenges. This could reflect limited exposure to Prolog concepts before the quiz or difficulty transitioning from imperative to declarative programming paradigms.

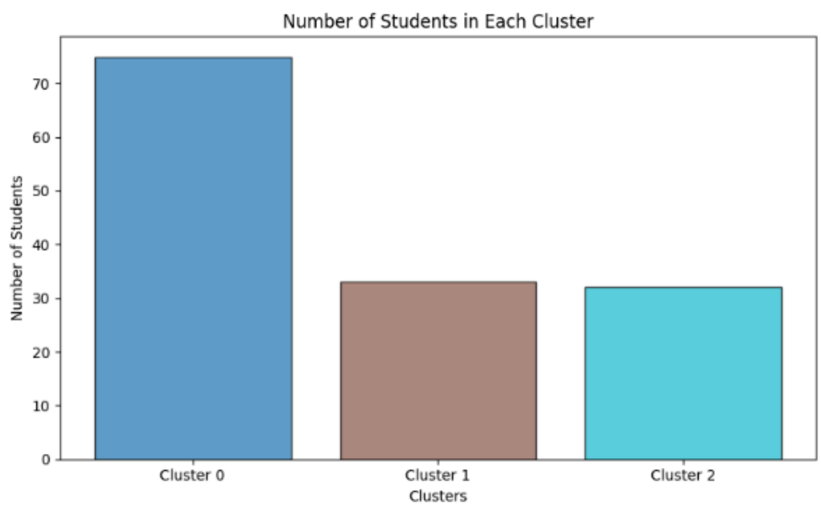

Figure 3. Cluster Size Distribution

The cluster size bar chart (Figure 3) provides a clear visualization of the distribution of students across the three clusters. The x-axis represents the clusters (Cluster 0, Cluster 1, and Cluster 2), while the y-axis indicates the number of students in each cluster. Cluster 0 contains the largest group of students, with over 70 participants, signifying that most students performed at an intermediate level during the quiz. Cluster 1 and Cluster 2 represent smaller groups, indicating fewer students achieved the highest (Cluster 1) and lowest (Cluster 2) performance levels.

The relatively smaller size of Cluster 1 suggests that advanced levels require greater cognitive effort and preparation, potentially reflecting the complexity of Prolog's advanced concepts. This distribution aligns with constructivist learning theories, emphasizing the role of scaffolding in helping students bridge gaps between basic and advanced learning stages. Cluster 0's dominance highlights a common transitional phase where students grapple with intermediate-level Prolog concepts. Outliers in this cluster may represent students who either excelled in foundational topics or struggled with advanced transitions. This disparity underscores the importance of individualized support during these transitional phases.

[1]Corresponding Author: Malakit L. Ram
[*]Corresponding Email: malakitr21@gmail.com

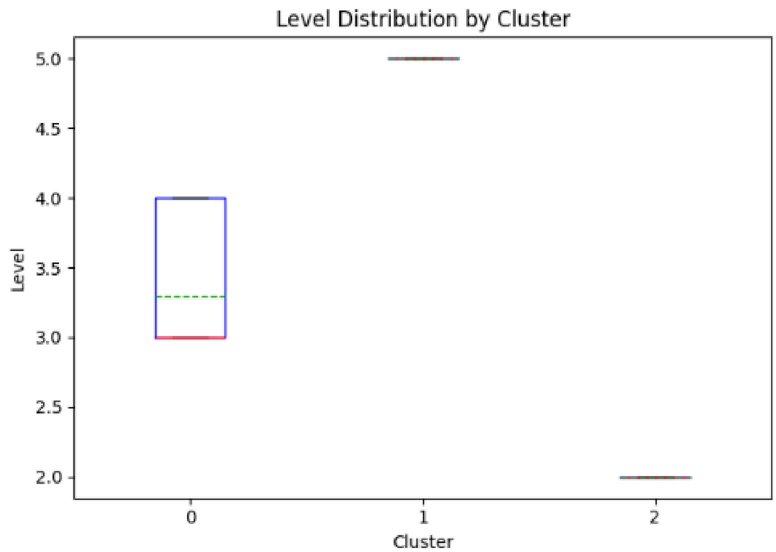

Figure 4. Level Distribution Across Clusters

The box plot (Figure 4) shows the distribution of levels achieved by students within each cluster, offering a deeper understanding of the spread and variability in their performance. The x-axis represents the clusters (Cluster 0, Cluster 1, and Cluster 2), while the y-axis represents the levels achieved by the students:

- **Cluster 0:** This cluster has a wider range of performance levels, with the box indicating the interquartile range (IQR) from approximately level 3 to level 4. The median (red line) lies close to level 3.5, suggesting a balanced distribution of mid-level performers.
- **Cluster 1:** This cluster is tightly grouped around level 5, with little to no variability, indicating that students in this cluster consistently reached the highest performance level within the given time limit.
- **Cluster 2:** Similar to Cluster 1, this cluster shows little variability but is centered around level 2, signifying students who struggled to progress beyond the initial levels.

The box plot highlights the performance consistency in Clusters 1 and 2, which consist of high and low achievers, respectively, while Cluster 0 shows greater variability, representing the majority of students navigating intermediate levels. This variability in Cluster 0 could be due to differences in students' prior exposure to Prolog or programming concepts. Moreover, the consistent performance in Clusters 1 and 2 may suggest that time constraints had a less pronounced impact on students at the performance extremes.

The results revealed several key observations and implications. Outliers in Cluster 0 indicated a subset of students with potential for higher achievement or unique learning strategies that distinguished them from their peers. This highlights the importance of identifying and supporting these students to help them reach

[1]Corresponding Author: Malakit L. Ram
[*]Corresponding Email: malakitr21@gmail.com

their full potential. The dominance of Cluster 0 also underscored the transitional challenges faced by most students as they navigated intermediate-level Prolog concepts. This finding emphasizes the critical need for structured guidance and scaffolding to aid students in overcoming these difficulties. Furthermore, external factors such as prior programming experience, familiarity with Prolog, and the 15-minute time constraint significantly influenced cluster placement, pointing to areas that warrant further investigation. These findings underscore the importance of tailored interventions to address diverse learning needs and adapt assessments to reduce external barriers, thereby supporting students at all performance levels.

Discussions

Key Findings

The findings of this study illustrate how K-Means clustering effectively grouped students based on their performance in the Prolog programming quiz, providing a clear framework for identifying learning patterns and informing targeted educational interventions. The analysis revealed three distinct clusters of students: those facing foundational challenges, those with moderate understanding, and a smaller group excelling at advanced levels. These results highlight the utility of integrating clustering techniques with adaptive learning tools, such as Prolog-based quizzes, in identifying areas where students require additional support, tailored interventions, or enrichment (Ahmed et al., 2020; Govender & Sivakumar, 2020; Ikotun et al., 2022).

The scatter plot (Figure 2) illustrates the distribution of students across clusters, with centroids representing the average performance within each group. Students in lower clusters (Cluster 2) struggled to progress beyond the initial levels, indicating difficulties in grasping basic Prolog concepts. On the other hand, students in higher clusters (Cluster 1) advanced more effectively, demonstrating proficiency in solving complex tasks. This disparity may reflect differences in prior programming knowledge, cognitive adaptability, and the impact of the 15-minute time constraint on learning progression. Constructivist learning theory suggests that foundational mastery is crucial for advancing in problem-solving tasks, aligning with findings from prior studies (Huang et al., 2021; Veneta Tabakova-Komsalova et al., 2023).

The bar chart (Figure 3) provides insights into cluster sizes, with most students falling into the mid-level performance cluster (Cluster 0). This reflects shared challenges in transitioning from basic to intermediate Prolog concepts. The smaller size of Cluster 1 suggests that advanced levels demand additional guidance, scaffolding, and exposure to higher-order problem-solving tasks, as emphasized in differentiated instruction frameworks (Hermenegildo et al., 2023).

The box plot (Figure 4) highlights performance variability, with Cluster 0 showing greater diversity in skill levels. This underscores the need for tailored instructional strategies to address varying levels of student preparedness. For example, differentiated approaches can help bridge performance gaps by emphasizing foundational reinforcement for struggling students and advanced problem-solving for proficient learners (Govender & Sivakumar, 2020; Sinaga & Yang, 2020).

[1]Corresponding Author: Malakit L. Ram
[*]Corresponding Email: malakitr21@gmail.com

Implications for Education

The clustering analysis provides several actionable recommendations to address the diverse learning needs of students. For students in lower clusters (Cluster 2), remedial activities and reinforcement of foundational Prolog concepts are essential to help them overcome consistent struggles and build a solid understanding of basic programming principles (Veneta Tabakova-Komsalova et al., 2023). Students in mid-level clusters (Cluster 0) would benefit from targeted interventions, such as guided practice and scaffolding, to support their transition from basic to intermediate topics, addressing the challenges commonly faced during this critical learning phase (Ahmed et al., 2020; Ikotun et al., 2022). For higher-performing students in Cluster 1, advanced problem-solving tasks and exploratory learning opportunities should be provided to deepen their understanding and foster independent learning (Hermenegildo et al., 2023).

These recommendations are consistent with broader educational theories that emphasize scaffolding, differentiated instruction, and personalized learning as effective strategies to support diverse learner needs. By implementing these approaches, educators can improve learning outcomes and reduce disparities across performance levels. Additionally, future research should explore advanced clustering techniques, larger sample sizes, and the incorporation of engagement metrics to further refine these strategies and enhance the effectiveness of adaptive learning systems.

This study contributes to the growing body of research in computational thinking and educational technology by demonstrating the potential of Prolog's declarative programming capabilities for adaptive assessments. The Prolog-based quiz program represents an innovative approach to curriculum design, offering dynamic question generation, real-time feedback, and structured progression. These features align with modern pedagogical goals of providing personalized, engaging, and scalable learning experiences.

The methodologies and tools developed in this study can be adapted for other programming courses, such as Python, Java, or C++, where progressive difficulty levels and real-time feedback can enhance learning. For example, clustering can be used to evaluate student engagement in courses on algorithm design or data structures, helping educators identify common challenges and adjust instructional strategies accordingly.

Future research should explore integrating Prolog-based tools into blended learning environments, incorporating advanced clustering techniques, and analyzing additional metrics like engagement and motivation. These efforts can further refine the effectiveness of adaptive learning systems and expand their applicability across diverse educational domains. By addressing these areas, educators and researchers can continue to leverage technology to improve learning outcomes and foster a deeper understanding of computational thinking.

[1]Corresponding Author: Malakit L. Ram
[*]Corresponding Email: malakitr21@gmail.com

Conclusion

This study demonstrated the effectiveness of K-Means clustering in analyzing student performance in a Prolog programming quiz, revealing distinct performance patterns: foundational challenges among lower-level learners, transitional difficulties for mid-level students, and advanced mastery among high achievers. These findings underscore the importance of tailored instructional strategies, including remedial activities for foundational concepts, targeted scaffolding for intermediate topics, and advanced problem-solving opportunities for proficient students.

The Prolog-based quiz program developed in this study leveraged Prolog's declarative nature to dynamically generate questions, provide real-time feedback, and enable structured, level-based progression. This adaptive and interactive assessment approach has significant potential for programming education and other domains that require evaluating student progression and engagement.

While the study was limited to a single quiz and employed the K-Means algorithm, which assumes spherical clusters, future research should address these limitations. Expanding the dataset, exploring alternative clustering techniques such as hierarchical or density-based clustering, and integrating additional metrics like engagement levels, learning behaviors, or time-on-task analysis could provide a more nuanced understanding of student performance.

Future studies could also explore the application of similar methodologies using other declarative programming languages, such as Haskell or SQL, to assess their utility in educational contexts.

For practical implementation, educators should consider integrating Prolog-based assessments into existing programming curricula, particularly in courses emphasizing logic and computational thinking. Key technical requirements include the availability of Prolog interpreters, seamless integration with learning management systems, and the capability to generate real-time feedback and adaptive assessments. Institutions should also develop targeted support programs that align with the identified performance clusters, addressing specific challenges faced by foundational, intermediate, and advanced learners.

Additionally, incorporating Prolog-based assessments into blended learning environments, such as online platforms and hybrid classrooms, could facilitate broader adoption and scalability. These tools could be extended to other programming courses, such as Python or Java, by adapting the structure and assessment methodology to suit the unique characteristics of these languages. By adopting these recommendations, educators and institutions can optimize instructional strategies, enhance the learning experience, and foster improved student outcomes across diverse educational settings.

References

Ahmed, M., Seraj, R., & Islam, S. M. S. (2020). The k-means Algorithm: A Comprehensive Survey and Performance Evaluation. *Electronics*, *9*(8), 1295. https://doi.org/10.3390/electronics9081295

Almansoury, F., Kpodjedo, S., & Boussaidi, G. E. (2022). Identifying Community-Supported Technologies and Software Developments Concepts by K-means Clustering. *International Journal of Advanced Computer Science and Applications*, *13*(6).

[1]Corresponding Author: Malakit L. Ram
[*]Corresponding Email: malakitr21@gmail.com

https://doi.org/10.14569/ijacsa.2022.01306108

Govender, P., & Sivakumar, V. (2020). Application of k-means and hierarchical clustering techniques for analysis of air pollution: A review (1980–2019). *Atmospheric Pollution Research*, *11*(1), 40–56. https://doi.org/10.1016/j.apr.2019.09.009

Hermenegildo, M. V., Morales, J. F., Lopez-Garcia, P., & Carro, M. (2023). Types, Modes and so Much More – The Prolog Way. *Lecture Notes in Computer Science*, 23–37. https://doi.org/10.1007/978-3-031-35254-6_2

Huang, S., Kang, Z., Xu, Z., & Liu, Q. (2021). Robust deep k-means: An effective and simple method for data clustering. *Pattern Recognition*, *117*, 107996. https://doi.org/10.1016/j.patcog.2021.107996

Ikotun, A. M., Ezugwu, A. E., Abualigah, L., Abuhaija, B., & Heming, J. (2022). K-means Clustering Algorithms: a Comprehensive Review, Variants Analysis, and Advances in the Era of Big Data. *Information Sciences*, *622*(622). https://doi.org/10.1016/j.ins.2022.11.139

Philipp Körner, Leuschel, M., Barbosa, J., Vítor Santos Costa, Dahl, V., Hermenegildo, M. V., Morales, J. F., Wielemaker, J., Diaz, D., Abreu, S., & Ciatto, G. (2022). *Fifty Years of Prolog and Beyond*. *22*(6), 776–858. https://doi.org/10.1017/s1471068422000102

Sinaga, K. P., & Yang, M.-S. (2020). Unsupervised K-Means Clustering Algorithm. *IEEE Access*, *8*, 80716–80727. https://doi.org/10.1109/access.2020.2988796

Veneta Tabakova-Komsalova, Stoyanov, S., Asya Stoyanova-Doycheva, & Lyubka Doukovska. (2023). Prolog Education in Selected Secondary Schools in Bulgaria. *Lecture Notes in Computer Science*, 144–153. https://doi.org/10.1007/978-3-031-35254-6_12

Warren, D. S. (2023). Introduction to Prolog. *Lecture Notes in Computer Science*, 3–19. https://doi.org/10.1007/978-3-031-35254-6_1

[1]Corresponding Author: Malakit L. Ram
[*]Corresponding Email: malakitr21@gmail.com

[1]Corresponding Author: Malakit L. Ram
[*]Corresponding Email: malakitr21@gmail.com