Impact in Computics

Impaxon

Article

An AI-Driven Game Analytics Framework for Intelligent Player Profiling in a Unity-Based Roll Ball Environment

Amanpreet Kaur [1,*], Amitesh Aggarwal [1], Neha Garg [1], Priyanka Datta [2]

[1] Chitkara University Institute of Engineering and Technology, Chitkara University, Punjab 140401, India
[2] Laxmi Institute of Technology, Sarigam, Gujarat 396155, India
* Correspondence: amanpreet.kaur@chitkara.edu.in (A.K.)

Abstract

Modern digital games generate extensive gameplay telemetry that can reveal patterns of player behavior and performance. This study proposes an AI-enabled game analytics framework for intelligent player profiling using gameplay data collected from a Unity-based Roll Ball environment. The Roll Ball game is implemented as an experimental platform consisting of three stages of ascending difficulties, such as Tier 1, Tier 2, and Tier 3, that are successively unlocked as the player passes from one stage to the next. While playing, the players interact frequently with the game environment, generating information related to their performances. The empirical analysis is performed by gathering information regarding the gameplay of the undergraduates. The results obtained from the scores of the items obtained during the gameplay are used to measure the performance of the player as they progress through the levels. To this end, two popular machine learning techniques, namely, Random Forest and Feedforward Neural Network (FNN), are adopted to assess the relation between the scores achieved in different levels of the Roll Ball game. The experimental dataset was split into a training set and a test set to examine the effectiveness of our proposed approaches. The experimental results obtained demonstrated the capability of machine learning algorithms in detecting performance-level related patterns from the gameplay data and hence obtaining insights about the player behavior under different levels of game difficulty. This work explores the possibility of employing machine learning with gameplay data collected through gameplay telemetry and aims to demonstrate the benefits of intelligent game analytics, player behavior analysis, and data-driven game design. Possible future works include integrating additional features such as time to complete levels, player movements, and interactions at each level to achieve a more detailed and robust modeling of the players and hence to improve the predictive power of our proposed approaches.

Keywords: Digital Game; Unity 3D Game Engine; Machine Learning; Classification; Random Forest; Feed Forward Neural Network

Citation: Kaur, A., Aggarwal, A., Garg, N., & Datta, P. (2026). An AI-Driven Game Analytics Framework for Intelligent Player Profiling in a Unity-Based Roll Ball Environment. *Impact in Computics* 2026, **2**, 2. https://doi.org/10.65500/computics-2026-002

Received: 02 January 2026 | Revised: 17 February 2026 | Accepted: 27 February 2026 | Published: 16 March 2026

1. Introduction

Video games have become really complex and interactive, with players constantly engaging with virtual objects, game mechanics, and environments that change all the time [1]. As players interact with these elements, they produce a huge amount of data that shows how they behave, perform, and make decisions [2]. This data is very helpful for developers of games because it tells them what

players like and don't like, and how to improve the game. Recent improvements in AI and ML have made it possible to automatically look at this information as well as learn a lot regarding how players act and behave. Game designers can use this information to find patterns in how players behave, which they may then utilize to make the game more enjoyable as well as challenging [3]. AI and ML can help game designers make games that are more based on data. This can make games better and players happier. Designers of games can use both gameplay data and advanced analytics to improve games and change how they are made [4].

Game analytics is a big area of study that focuses on collecting and analyzing gameplay data to learn more about how players interact with digital games. Through analyzing data from telemetry, it is possible to detect patterns in players' behavior, evaluate their performance, and assess the effectiveness of the game's operation. Keeping this information in mind, it will be easier to create an adaptive gaming system that can change its parameters based on some factors, individualize the gaming experience, and hold the attention of players. A key component of game analytics is player modeling, which aims to construct computational representations of player characteristics such as skill level, gameplay proficiency, and interaction patterns.

Machine learning is a research field in the broader context of player modeling, and using machine learning techniques, the data of player behavior in the game can be analyzed automatically and more efficiently. Using supervised learning, the hidden characteristics of player behavior can be learned, and players can be classified into different skill levels according to their performance in gameplay actions and their corresponding outcomes [4]. The skill-level classification of players can provide very valuable information to the designers of a game to understand the performance differences and progression levels of players in a virtual game world. In this respect, using machine learning-based player modeling approaches, more intelligent game environments can be designed.

With the new game development engines emerging, such as Unity 3D, the game developers can now efficiently and quickly build a game level environment and numerous activities within it, and collect relevant gameplay data or telemetry [5]. Unity 3D provides real-time graphics rendering, realistic physics simulation, and other features such as scripting certain parts of the game level environment with C# code [6]. In this work, the study shows that Unity 3D can also be used to design experimental game levels for research purposes and as a testing ground for ML algorithms for the purposes of gaming analytics.

In this study, a Roll Ball game is developed using the popular game engine Unity, to be used as a controlled environment to gather gameplay data from users. In the game, the user controls a rolling ball to collect crystals, located at different stages and of varying difficulty levels [5]. The system computes the score of the action made by the user, according to the performance achieved at each stage of the game environment. The collected gameplay data were prepared by applying feature engineering techniques and then analyzed by employing machine learning approaches to model and assess the user performance.

Although several studies have explored gameplay analytics and player modeling, many of them rely on complex commercial game datasets or proprietary telemetry systems. Limited research has explored lightweight experimental environments that integrate gameplay telemetry with machine learning models for player skill classification. Therefore, this study proposes an AI-enabled game analytics framework implemented using a Unity-based Roll Ball environment to investigate player profiling through supervised learning techniques.

The purpose of this study is to develop a game analytics framework based on Artificial Intelligence (AI) for intelligent player profiling. The framework proposed in this study will have the capability of collecting the gameplay data, feature extraction, and applying a machine learning algorithm in order to recognize the patterns in the data. In order to classify the players into suitable skill levels based on their performance, two supervised learning algorithms, Random Forest and Feedforward Neural Network (FNN), will be applied. The proposed framework will be used to investigate the relationship between the scores received from the gameplay and the player's progress in the digital games, and also apply machine learning in order to classify and analyze the player behavior in the digital games [7].

The remainder of this paper is organized as follows. Section 2 presents a review of related research on game development, gameplay analytics, and machine learning applications in gaming environments. Section 3 describes the tools and technologies used for developing the Roll Ball game environment and collecting gameplay data. Section 4 explains the methodology and system architecture of the proposed AI-enabled game analytics framework. Section 5

presents the experimental validation and performance analysis of the implemented machine learning models. Finally, Section 6 concludes the paper and outlines potential directions for future research in intelligent game analytics.

2. Related Work

Game analytics and player modeling have been of great interest because of the growing access to gameplay telemetry information. Scientists have investigated different methods of studying player behavior, performance, and interaction patterns to improve the game design and player interaction.

Several research studies have concentrated on enhancing game development procedures and tools. Research conducted by Casamayor et al. [8] on automated bug localization methods in video games based on a genetic algorithm showed greater effectiveness in detecting faults in complex gaming systems. In a similar way, Marin-Lora et al. [9] introduced a simplified scripting method to make game development less complex and more accessible to developers. Fleck et al. [10] proposed an assistant to edit and analyze game content in Unity environments, which facilitated debugging, testing, and analysis of gameplay. These research papers underscore the need of strong development environments that enable effective game development and data management. XR technologies, including AR, VR, and MR. The study identified the problems involved in testing XR applications due to their interactive aspect and six degrees of freedom [11]. The importance of using testing tools was stressed in this paper. Matthew S. Willsey et al. [12] described the use of Brain-Computer Interface (BCI) technologies to facilitate natural interactions between paralytic individuals and virtual as well as physical worlds through deliberate movement of fingers.

Moreover, studies have highlighted the significance of gameplay data in the study of player interactions. Unity and other game engines offer a versatile framework to create and record gameplay telemetry and analyze it through a structured analysis of player behavior. Kang et al. [13] investigated the visualization of big data with the use of an interactive environment the application of which assists in the interpretation of complex behavioral data. Ullmann et al. [14] suggested ways to analyse game engine designs, leading to a better insight into the components of the system and their interaction. These methods aid the examination and understanding of gameplay data of contemporary gaming systems. Research on the effects of video games among the youth and found that there was improvement in terms of social interaction skills, decision making, and stress relief [15]. Researcher examined the use of digital technology in the form of mobile applications, video games, and robots for increasing the emotional and cognitive capabilities of children suffering from ADHD [16]. Vicente Jover et al. [17] showed how metaverse and unity virtual technology can be applied.

One of the most important developments in player modeling and game analytics has been machine learning. It allows automatic identification of patterns in the gameplay data and allows for classifying players by their performance and behavior. The supervised learning methods are quite common in determining the relations between the metrics of gameplay and the level of player skills, so that it is possible to create adaptive gaming systems and data-driven gaming systems. These techniques are essential in improving player experience in terms of smart analytics and customization of games.

Nevertheless, the available literature is mostly on game development tools, visualization techniques, or systems-level analysis, with little attention on the combination of gameplay telemetry and machine learning methods to classify players in terms of skills in a controlled experimental setting. Moreover, several research works are based on complicated or proprietary data, which limits reproducibility and experimental validation.

Thus, the research suggests an AI-powered game analytics system with a Unity-based Roll Ball setting to gather gameplay data in a systematic manner and use machine learning algorithms, i.e., Random Forest and Feedforward Neural Network (FNN), to profile and classify intelligent players.

3. Tools and Technologies

This study aims to develop a system that interacts with a game environment and machine learning tools. The core elements of this system are game mechanics for gathering the required data for the functions that the system will perform, and an AI engine for the classification of players through the analysis of their behavior and attributes. The game environment used in this study is Roll Ball. By analyzing the gameplay data of the players with the machine learning algorithms, the data was examined for the validity of the system.

3.1 Gameplay Environment

The Roll Ball game environment was developed using the Unity game development platform, as illustrated in Figure 1. Unity is a real-time game engine that supports

game development with 3D graphics and event-driven programming through scripts. The functions of the environment used as an experimental environment are as follows: The players interact with the objects in the game environment by performing tasks like moving through levels and collecting the crystals.

The primary purpose of the game environment is to generate gameplay telemetry reflecting player performance. Every time a player interacts with something, it adds to their score, which is the main way to tell what level they are. So, instead of focusing on graphics design, the environment is used as a supervised platform for gathering structured gameplay data.

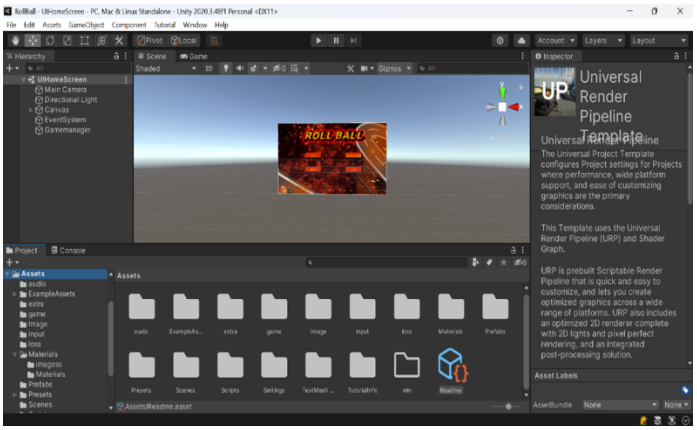

Figure 1. Unity Game Engine

The gameplay mechanics and the telemetry logging were implemented via C# scripts [18]. As shown in Figure 2 and Figure 3, the scripts collected the information of the player's behavior and performance, such as the score the player gets in the end. The scripting layer provided an automated method of data collection, and it is used in the machine learning-based player analysis.

Figure 2. GameObjects(1)

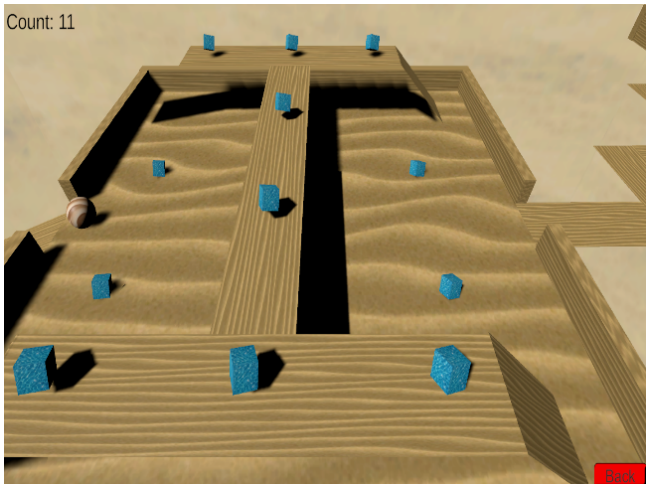

Figure 3. GameObjects(2)

3.2 Gameplay Data Collection

Data associated with gameplay is generated when the user performs actions on the elements of the Roll Ball game. The score obtained by the user at the end of each level is registered for each occasion the game level is played [19][20].

This dataset contains scores of various players. The scores are quantitative values that measure the performance of a player in various aspects of the game. These scores can be used to create the player performance profiles. The performance profiles are then used as training data for the machine learning classifier in order to assign the players to various skill groups [21].

3.3 Machine Learning Framework

Machine learning techniques are applied to analyze the gameplay dataset and perform player classification. Two supervised learning models are implemented:

1. Random Forest Classifier
2. Feedforward Neural Network (FNN)

Random Forest is an ensemble learning method that trains many decision trees to classify samples quickly and avoid overfitting. A feedforward neural network is a deep learning model that can train samples to predict the output efficiently by learning non-linear features between the samples via multiple hidden layers.

Both models are trained on the gameplay dataset and classify a player into a number of predefined skill tiers based on their player score. The models learn how the score relates to each skill tier and classify a player into a skill tier. The models can be used to automatically classify players into a suitable skill tier. The decision trees used in the Random Forest model were set to 100 decision trees, default depth, and splitting criteria to maintain a consistent performance. The Feedforward Neural Network is two-

layered with 16 neurons in each layer, with ReLU activation functions.

4. Methodology

This study proposes an AI-enabled player profiling framework that integrates gameplay interaction with machine learning analytics. The Roll Ball game environment serves as a controlled experimental platform where players perform gameplay activities that generate measurable performance data. These data are subsequently analyzed using machine learning techniques to classify players into different skill levels.

The Roll Ball system development and evaluation process consists of three stages: Pre-Production, Production, and Post-Production, which are responsible for designing the game environment, collecting player performance data, and implementing the machine learning algorithm that classifies the players, as illustrated in Figure 4.

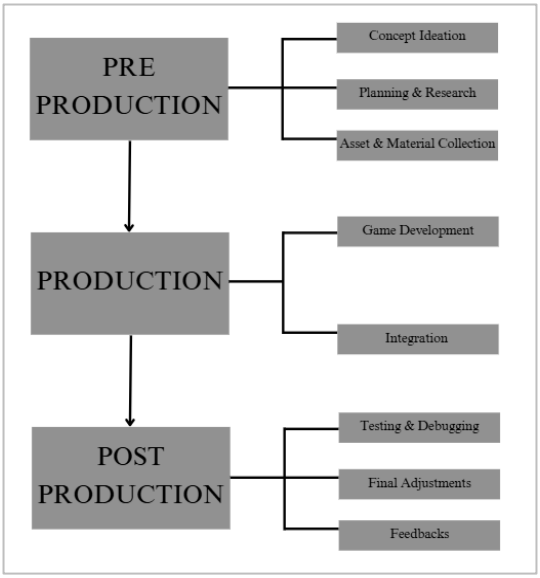

Figure 4. Phases of development

4.1. Phase 1: Pre-Production

The pre-production phase is all about coming up with ideas, making plans, and designing the system. The main goal at this point is to figure out how the game will work and set up the system needed to collect measurable gaming data. The idea behind the Roll Ball game is to have participants control a rolling ball as well as collect the crystals that are spread out over the game world. The participant's score, which is the main way to profile players, goes up with each crystal they collect.

The research design includes research and planning activities of gaming structure, data collection requirements, and machine learning integration, among other conceptual planning. The framework maintains records of each participant's total score, which allows it to create custom gaming telemetry data that could be used in regular ML studies.

This is also when the game's assets and resources, such as level parts, gameplay objects, and visual elements, are found and set up. The main goal of this study is to create gameplay interactions that produce structured behavioral data for player analytics, which is different from traditional game development methods that focus on how the game looks.

4.2. Phase 2: Production

The main objective of the production phase is to create and put together the system that makes the game work. Now, the Roll Ball game framework is being built with the Unity game design engine. You can use simple scripts to make 3D models and modify and add standard gameplay logic on this platform. Participants move a ball around the levels of the game and pick up crystals that raise their score. This is how they really interact with the world around them. These trades give players numbers that tell them how well they are doing in the game. The main part of the game is shown in Figure 5, in which the player communicates with the surroundings and acquires crystals to raise their score.

The game itself has three different themed areas that help to make it more fun to play. Figure 6 shows the Fire environment, which has lava-themed parts; Figure 7 shows the Ice environment, which has snowy parts and icy surroundings; and Figure 8 shows the Desert scene, which has dry land and desert-themed materials. The fundamental concepts behind the gameplay stay consistent on all levels, even though the graphics change.

This indicates that the total score for each player is calculated based on how well they performed, rather than how varied the environments are.

Figure 5. Production (1)

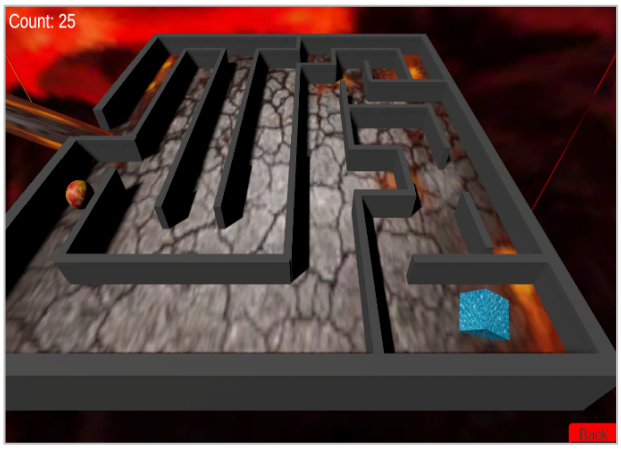

Figure 6. Production (2)

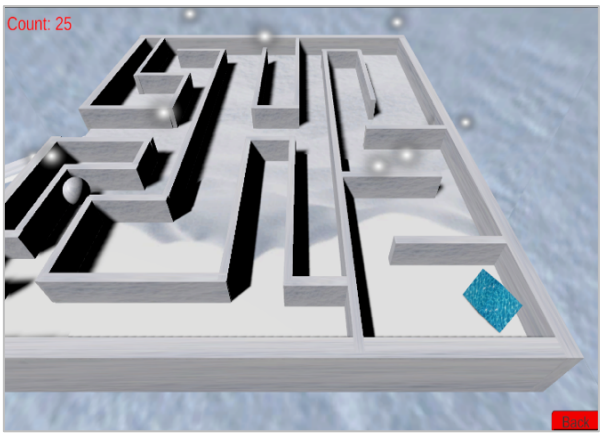

Figure 7. Production (3)

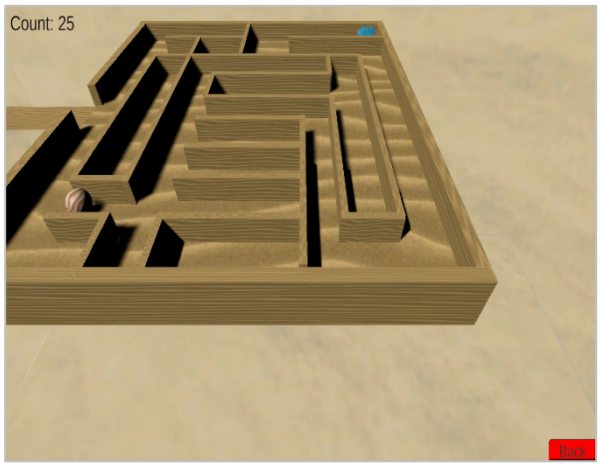

Figure 8. Production (4)

Adding gaming mechanics, scoring systems, and data logging functionality throughout this phase makes it possible to make reliable gameplay telemetry. Then, this telemetry is used to make the dataset for machine learning research and game player modeling.

4.3. Phase 3: Post-Production

Testing, identifying and fixing bugs, analyzing, and collecting gameplay data are all part of the post-production phase. This is when the Roll Ball system that was designed is tested in order to make sure that the game runs smoothly and that players' scores are recorded appropriately. The debugging process is done to fix bugs and make sure the game runs well.

Once the system is validated, 100 undergraduates play the game and try to finish the levels while collecting crystals. The system maintains records of every player's total overall score, which tells them how effectively they performed in the game, depending on how they interacted with other players. After testing the game, small changes are made to improve the system and make sure that gameplay information is recorded correctly.

Feedback from standard game activities is also utilized to enhance the experience and ensure the effectiveness of the data collection process. The dataset acquired serves as the foundation for the machine learning-driven player classification procedure in the subsequent analytical phase.

4.4. System Architecture of AI-enabled Game Analytics

The suggested system uses a multi-phase analytical pipeline that combines standard gameplay telemetry collection alongside machine learning techniques to make smart player profiling as well as performance analysis possible. The architecture changes how people play games into structured information that machine learning models can use to learn from. The system architecture consists of five main stages, as illustrated in Figure 9.

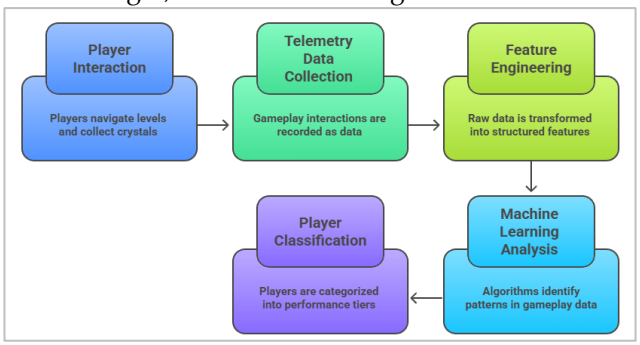

Figure 9. Multi-stage analytical pipeline

4.4.1. Player Interaction

The players engage with the Roll Ball game design by moving through hard levels and collecting crystals in the first stage. These reactions are examples of how players interact with each other in ways that affect their performance in the game. As you play the Roll Ball game, it gets harder, as shown in Figure 10:

- Tier 1 – Beginner Level
- Tier 2 – Intermediate Level
- Tier 3 – Advanced Level

Figure 10. Levels of the Game

Players have to finish each level before they can move on to the next one. This makes the game harder over time. Higher scores usually mean that you played better and had more control over the game's mechanics.

4.4.2. Telemetry Data Collection

Gameplay mechanics are applied to capture all the events in the game. The dependent variable, in regard to telemetry, in this specific study, will be the score of the participant, which indicates how well the player was able to collect the crystals.

4.4.3. Feature Engineering

Feature engineering transforms unstructured gameplay telemetry into structured variables, which can be utilized in machine learning evaluations. The gameplay data generated in this research has two major attributes as shown in Table 1.

Table 1. Description of Gameplay Dataset Attributes

Attribute	Description
Player Score	Total score obtained during gameplay
Game Level (Tier)	Difficulty level reached or completed by the player

These organized attributes are a reflection of the player's performance and development in the gameplay environment. Pre-processing of the data set was done by putting the gameplay scores and gameplay levels into a structured format, and basic normalization was done where necessary.

4.4.4. Machine Learning Analysis

The machine learning algorithms seek patterns in the behavior and performance of players and the performance during gameplay in the processed dataset. In this paper, two supervised learning algorithms have been used:

- Random Forest Classifier
- Feedforward Neural Network (FNN)

These algorithms compare the score on a game to the degree of progress so as to discover patterns in behavior.

4.4.5. Player Classification

At last, the models that were trained put the players into distinct performance tiers based on how well they do in the game. To test how well the model works, the dataset is split into two parts: training (80%) and testing (20%). To see how well the models work, this study uses standard classification metrics like precision, recall, and F1-score.

The proposed framework makes it possible to automatically profile players and analyze gameplay using data through this classification process.

5. Validation

The present analysis is performed on one train-test split; hence, the performance could change according to various data splits. The inclusion of cross-validation would give more consistent and confident estimates of the performance. Two supervised machine learning models, a Random Forest and an FNN, were used to test how well the proposed AI-driven player profiling framework worked. The models are trained using data obtained from the Roll Ball game. The dataset contains player scores and the player tiers that go with them, which show how good each player is.

5.2. Random Forest

Random Forest is an ensemble learning algorithm that constructs numerous decision trees and then combines the solutions each decision tree gives to come up with a final categorized solution. This algorithm is based on bootstrapping accumulation (bagging), i.e., the decision trees are learned on a random subset of the data set. Such an approach causes the model to be more stable and is less prone to fitting too well. The random Forest model used in this study was trained to classify players into three skill levels: Tier 1, Tier 2, and Tier 3. This was pegged on their effectiveness in playing. This algorithm was applied to 100 decision trees to determine the quality of players and the speed with which they grow.

5.2.1. Random Forest Model Result

The dataset under test was processed using the Random Forest model, which predicted the level of each player [22]. The model used the relationships between players' skill levels and their gameplay scores to figure out how to put them into different tiers. Figure 11 shows how the Random Forest model predicted the different player tiers. There were 37 players in Tier 1, 31 players in Tier 2, and 29 players in Tier 3. Three players were put in the

"Unclassified" group because their scores didn't fit neatly into any of the defined tier ranges. This distribution shows that the Random Forest model was able to put most players into useful skill groups, which made it easier to look at how players did and acted in the game.

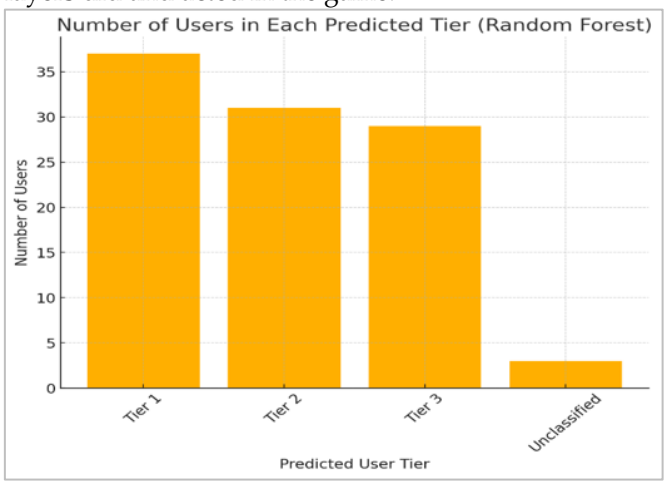

Figure 11. Distribution of player classification results obtained using the Random Forest model.

5.2. Feedforward Neural Network (FNN) Methodology

A Feedforward Neural Network (FNN) is a type of architecture of deep learning that processes information in order through many layers of neurons that are all connected. The input layer is where data goes into networks. After that, the data goes through layers that are hidden, where it learns and processes patterns. The data finally gets to the outer layer, which makes the final prediction. FNN doesn't have any loops or cycles. Many layers of neurons work together to make the result. The network changes the weights to make the prediction more accurate. This process goes on as long as the system makes a perfect prediction [23]. The model was also trained on a standard optimizer (Adam) with a fixed learning rate and trained on several epochs to make sure the convergence.

For the RollBall game, a FNN model is used in the RollBall game to sort players into groups based on their scores and skills. It has two hidden layers, and each layer has sixteen neurons that work on the data. The Rectified Linear Unit (ReLU) Activation Function helps the network learn more. The dataset is also split into two parts: 80% for training and 20% for testing.

1. FNN Model Result

After training, the FNN put players into three gameplay tiers based on their scores. A pie chart was used to show the classification results, which showed how players were spread out across the different skill levels. Figure 12 shows that the neural network can learn patterns from how well players do in games and use those patterns to put players into the right tiers.

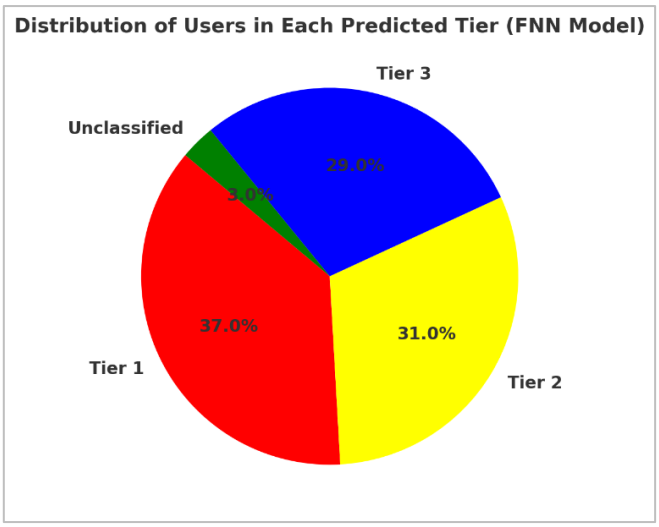

Figure 12. FNN Model Result

5.3. Comparison of Random Forest and Feedforward Neural Network (FNN) Models

1. Methodology and Approach

Both the Random Forest and Feedforward Neural Network models are trained to differentiate users based on their scores in the RollBall game. Although both models are used for the same purpose, their working is different from each other. In the Random Forest, multiple decision trees are created, and they evaluate the result to make accurate predictions. Whereas in FNN, data is processed through multiple layers to combine the final result with accurate predictions.

2. Accuracy and Classification Performance

The evaluation results show that both models are capable of learning patterns from gameplay data and performing player classification. However, differences in performance can occur depending on the complexity of the dataset and the ability of each model to capture underlying patterns.

3. Graphical Representation and Insights

The stacked bar chart is used to compare the final results of the Random Forest and FNN models, as shown in Figure 13. The orange bar represents the classification performed by Random Forest, and the Maroon bar represents the classification done by the FNN model. The bar appears uniform because both models provide similar predictions.

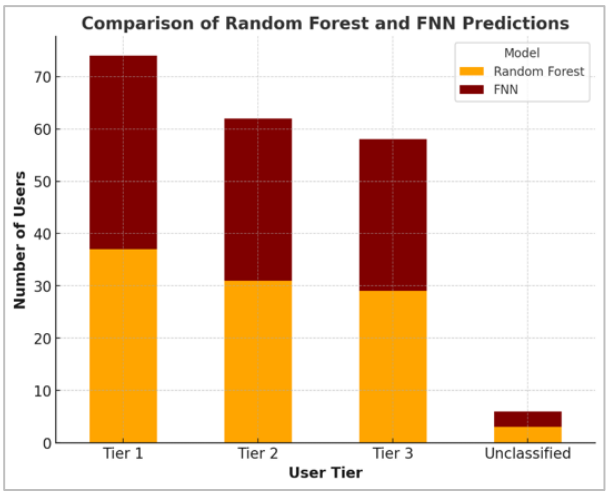

Figure 13. Comparison of Random Forest and FNN Models

4. Model Selection Considerations

Both methods perform equally well, but their functionality differs.

- Decision-making in Random Forest is easy to understand. The data processing is quick, and there is no requirement for high computing power.
- The Feedforward Neural Network model can handle large datasets. It is better for complex patterns. FNN requires higher computational resources due to its deep learning architecture.
- Adding the number of tuning in FNN enhances performance and accuracy.

5. Outcome

Since both models provide the same results, either can be used for the task with full confidence. Random Forest is suitable for scenarios requiring clarity and quick processing. Feedforward Neural Networks are preferable for deep learning and flexibility. The stacked bar graph proves that Both models demonstrate the feasibility of applying machine learning for gameplay analytics; however, the current feature set limits classification performance, both models demonstrate the feasibility of applying machine learning for gameplay analytics; however, the current feature set limits classification performance. A comparative summary of existing approaches and the proposed method is presented in Table 2.

The comparison points out that the proposed framework provides a more structured, reproducible, and application-oriented framework than the existing studies. It particularly stresses player-based analytics and real-time model assessment, which is appropriate in adaptive and intelligent gaming settings.

Table 2. Comparison of Related Works with the Proposed Method

Feature	**Existing Approaches (Related Work)**	**This Work (Proposed AI-enabled Framework)**
Primary Focus	Frequently specialized in game development tools or visualization techniques or system-level analysis.	Concentrates on combining gameplay telemetry with machine learning to classify the skill of players automatically.
Data Source	A large number of them depend on multifaceted business data or internal telemetry.	Plays in a lightweight, controlled experimental setting (Roll Ball game) created in Unity 3D.
Accessibility	Limited reproducibility because of proprietary data and complicated structures.	Created as a reproducible structure of academic research and testing of ML algorithms.
Methodology	Frequently includes automated localization of bugs or development scripting.	Has a multi-phase analytical pipeline: Player Interaction → Telemetry Collection → Feature Engineering → ML Analysis.
ML Models Used	Several, though not necessarily directly compared, in a single experimental system.	Compares Random Forest and Feedforward Neural Networks (FNN) directly, on the same classification task.
Outcome Goal	The overall enhancement of the game design or visual representation of data.	Active profiling to definite performance levels (Tier 1, 2, 3) to permit adaptive gaming systems.

The comparison points out that the proposed framework provides a more structured, reproducible, and application-oriented framework than the existing studies. It particularly stresses player-based analytics and real-time model assessment, which is appropriate in adaptive and intelligent gaming settings.

5.4. Confusion Matrix for Roll Ball Classification

Figure 14 shows the distribution between actual and predicted classifications, giving an insight into how the model distinguishes between the Tier 1, Tier 2, and Tier 3 players.

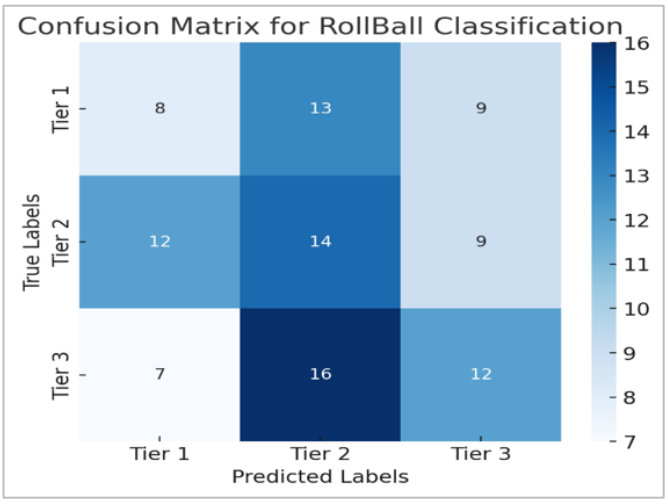

Figure 14. Confusion Matrix for Roll Ball Game Classification

Table 3 demonstrates that the classification provides information such as precision, recall, and F1 score of each of the tiers. Precision informs you about how many of the expected players in a tier are actually in that tier, and also informs you how well the model explains all the real cases of each tier. The F1-score, a trade-off between recall and precision, is the overall performance of the classification. The accuracy of the model is approximately 34, that is, the model has difficulty in classifying items properly. It might require an improved choice of features, additional training information, and enhanced optimization of hyperparameters. Making these parameters better can help make the Roll Ball game's classification model more reliable and stronger.

Table 3. Performance Metrics of the Roll Ball Game Classification Model

Class	Precision	Recall	F1-Score	Support
Tier1	0.296296	0.266667	0.280702	30
Tier 2	0.325581	0.4	0.358974	35
Tier 3	0.4	0.342857	0.369231	35
Accuracy	0.34	0.34	0.34	0.34
Macro Average	0.340626	0.336508	0.336302	100
Weighted Average	0.342842	0.34	0.339082	100

The confusion matrix and performance metrics demonstrate how the proposed player profiling framework can sort things out. The overall effectiveness of the model to classify things was about 34%. This implies that the models will be able to identify general tendencies in performance, but the features at this point are not good enough to make better predictions. This result shows how important it is to add more gameplay features to improve player modeling. Such aspects as the time needed to complete a level, the movement of the players, the frequency with which the players talk to each other, and their decision-making process might provide better representations of the behavior of the players and make the classification of the players more accurate. The results indicate that it is possible to apply machine learning techniques to the gameplay telemetry to both proactively profile players and analyze their performance, even though the accuracy is moderate.

6. Limitation

Despite the fact that this research proves the possibility of using ML algorithms for player profiling, there are some constraints that can be highlighted as well.

To begin with, the current research relies on the 100-sample dataset. Such small amounts of input data will definitely have some limitations as far as the learning process is concerned, especially when talking about a deeper analysis performed by a deep learning algorithm, like a feed-forward neural network (FNN).

Secondly, the set of features to classify players into different categories is rather limited and includes only two characteristics: players' scores and game level. Even though these features may be used to describe a player's performance to some extent, it is definitely not enough to understand player behavior. The lack of more developed gameplay mechanisms like completion time, movement patterns, how often they interact, and the strategies to choose behaviours impairs the predictive capacity and richness of behavioural analysis.

Also, the results of classification of both the Random Forest and FNN models were similar, indicating that the formulation of the problem at hand is not maximizing the potential of the modern machine learning methods. Simpler rule based or baseline models may be able to lead to similar results in such a case.

Lastly, the controlled Roll Ball environment, although convenient in structured data collection, might not be as reflective of the gaming conditions in the real world as it is. This restricts the relevance of the results to more dynamic and larger-scale gaming settings.

7. Conclusion

This paper provided an AI-based game analytics system to profile players intelligently based on gameplay telemetry data in a Unity-based Roll Ball system. The suggested framework incorporates gameplay interaction, data collection, feature engineering, and machine learning

analysis to categorize the players into various levels of skill. Two supervised learning models, Feedforward Neural Network (FNN) and Random Forest, were used to estimate the correlation between gameplay scores and player progression. It allows for classifying the performance of players and giving an opportunity to classify the players. Still, all the above-mentioned approaches can be described as rather ineffective when it comes to working with such a problem since the data used for performing such a classification is rather limited. However, the present study indicates that the use of gaming telemetry in intelligent player modeling and games that are adaptable is quite feasible. To sum up, the application of gaming telemetry allows gathering some valuable information on players' performance and behavior, which might be helpful in improving players' engagement with gaming and personalizing the gaming experience for each individual player. As to the improvement of the situation, the following steps can be proposed in the future, including adding more participants, including more behavioral metrics as independent variables, cross-validation, application of the methods of statistical analysis, the construction of the baseline models based on the Logistic Regression, SVM, and KNN.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data associated with this study are available from the corresponding author upon request.

Conflicts of Interest: The authors declare no conflict of interest.

References:

1. Granic, I., Lobel, A., & Engels, R. (2014). The Benefits of Playing Video Games. The American Psychologist, 69, 66-78. https://doi.org/10.1037/a0034857.
2. de Aguilera, M., & Mendiz, A. (2003). Video games and education. *Computers in Entertainment, 1*(1), 1–10. https://doi.org/10.1145/950566.950583.
3. Bavelier, D., Green, C. S., Han, D. H., Renshaw, P. F., Merzenich, M. M., & Gentile, D. A. (2015). Brains on video games. *Nature Reviews Neuroscience, 12*(12), 763–768. https://doi.org/10.1038/nrn3135
4. Hu, C. (2024). Research on the integrated application of machine learning in Unity. *Applied and Computational Engineering, 82*(1), 161–166. https://doi.org/10.54254/2755-2721/82/20241033
5. Prot, S., McDonald, K. A., Anderson, C. A., & Gentile, D. A. (2012). Video games: Good, bad, or other? *Pediatric Clinics of North America, 59*(3), 647–658. https://doi.org/10.1016/j.pcl.2012.03.016
6. Shaffer, D. W., Squire, K. R., Halverson, R., & Gee, J. P. (2005). Video games and the future of learning. *Phi Delta Kappan, 87*(2), 105–111. https://doi.org/10.1177/003172170508700205
7. Jagli, D., Chandra, S., Dhanikonda, S. R., & Laxmi, N. (2024). Artificial intelligence usage in game development. *SSRN Electronic Journal*. https://doi.org/10.2139/ssrn.4929567
8. Casamayor, R., Arcega, L., Pérez, F., & Cetina, C. (2025). Boosting bug localization in software models of video games with simulations and component-specific genetic operations. *Software and Systems Modeling*. https://doi.org/10.1007/s10270-024-01253-2
9. Lora, C. M., & Chover, M. (2025). GameScript: A simplified scripting language for video game development. *Multimedia Systems*. https://doi.org/10.1007/s00530-024-01574-8
10. Fleck, P., Hochortler, M., Kastl, D., Gotschier, G., Pirker, J., & Schmalstieg, D. (2025). CECILIA: A toolkit for visual game content exploration and modification. *IEEE Transactions on Games*. https://doi.org/10.1109/TG.2025.3528513
11. Gu, R., Rojas, J. M., & Shin, D. (2025). Software testing for extended reality applications: A systematic mapping study. *arXiv*. http://arxiv.org/abs/2501.08909
12. Willsey, M. S., et al. (2025). A high-performance brain–computer interface for finger decoding and quadcopter game control in an individual with paralysis. *Nature Medicine, 31*. https://doi.org/10.1038/s41591-024-03341-8
13. Kang, K. L., et al. (2025). Creating informative experiences through a visual and interactive representation of health and social care data. *Information Visualization*. https://doi.org/10.1177/14738716241309243
14. Ullmann, G. C., Guéhéneuc, Y. G., Petrillo, F., Anquetil, N., & Politowski, C. (2025). SyDRA: An approach to understand game engine architecture. *Entertainment Computing, 52*. https://doi.org/10.1016/j.entcom.2024.100832
15. Babanova, A., Ibragimova, Z., Mubarakshin, A., Khaziev, M., & Uboitseva, E. (2025). Video games as cultural tools for youth empowerment and activism development. *Journal of Lifestyle and SDGs Review, 5*(1), 1–13. https://doi.org/10.47172/2965-730X.SDGsReview.v5.n01.pe04200
16. Doulou, A., Pergantis, P., Drigas, A., & Skianis, C. (2025). Managing ADHD symptoms in children through the use of various technology-driven serious games: A systematic review. *Multimodal Technologies and Interaction, 9*(1), 1–26. https://doi.org/10.3390/mti9010008
17. Jover, V., Sempere, S., & Ferrándiz, S. (2025). The Creation of Virtual Stands in the Metaverse: Applications for the Textile Sector. *Electronics*, *14*(2), 359. https://doi.org/10.3390/electronics14020359
18. Ali, A. M. (2023). An Empirical study of dependency injection lifetime effects in C# applications. *Zenodo (CERN European Organization for Nuclear Research)*. https://doi.org/10.5281/zenodo.14613867
19. A. C. Vidal, Development of A Survival Game in A Unity3d Environment, BSc thesis work, Electrical Engineering and Informatics, Budapest University of Technology and Economics, Budapest, Hungary, 2017
20. Hillaire, S. (2020). A scalable and production ready sky and atmosphere rendering technique. *Computer Graphics Forum*, *39*(4), 13–22. https://doi.org/10.1111/cgf.14050
21. Zhang, B., & Hu, W. (2017). Game special effect simulation based on particle system of Unity3D. In *Proceedings of the 16th IEEE/ACIS International Conference on Computer and Information Science (ICIS)* (pp. 595–598). https://doi.org/10.1109/ICIS.2017.7960062

22. Farnaaz, N., & Jabbar, M. A. (2016). Random forest modeling for a network intrusion detection system. *Procedia Computer Science, 89*, 213–217. https://doi.org/10.1016/j.procs.2016.06.047

23. M. A-Mudhaf, A feed forward neural network approach for matrix computations, degree of Doctor of Philosophy, Department of Mechanical Engineering, Brunel University of London, UK, 2001