International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 09 Issue: 08 | Aug - 2025 SJIF Rating: 8.586 ISSN: 2582-3930

Player Action Prediction System using Machine Learning

[1] Akash R, [2] Dr. Shankaragowda B B
[1]Student,4th semester, Department of MCA, BIET, Davanagere, India
[2]Associate Professor,HOD, Department of MCA, BIET, Davanagere, India

ABSTRACT

In competitive gaming and sports analytics, understanding and anticipating player behavior is key to gaining strategic advantages. The **Player Action Prediction System** uses machine learning to predict a player's next move based on historical gameplay data, decision patterns, and environmental context. This system is beneficial in applications such as e-sports coaching, game AI enhancement, and player behavior analytics. Using supervised learning models like Random Forest, Decision Trees, Support Vector Machines (SVM), and deep learning (LSTM for sequential prediction), this system analyzes gameplay logs including actions, timestamps, game states, and player metrics. The project includes data preprocessing, feature engineering, training of models, and deployment in a prediction environment. The system outputs the most probable next action a player will take, which can be used by game engines or trainers to make real-time decisions.

INTRODUCTION

Player behavior analysis has gained massive importance in modern gaming and sports technologies. Predicting a player's next action allows systems to design adaptive AI, balance games, prevent cheating, and offer training insights in e-sports. Traditional systems depend on static heuristics and lack real-time adaptability. This project aims to build a dynamic and intelligent player action prediction system using machine learning. The model will learn from player movements, previous choices, and game situations to make accurate next-action predictions. The focus is on providing accurate, real-time, and explainable outputs that can be integrated into gaming systems or coaching tools.

LITERATURE REVIEW

The prediction of player actions has become an essential aspect of modern gaming, sports analytics, and AI development. Player Action Prediction (PAP) systems aim to forecast a player's future moves by analyzing historical gameplay data, offering benefits such as enhanced gaming experiences, adaptive AI opponents, and improved decision-making in sports. Machine learning (ML) techniques have proven to be invaluable for these tasks, enabling the identification of complex patterns and enabling accurate predictions in real-time.

Early efforts in player modeling were often based on rule-based or heuristic systems. These approaches were limited in their flexibility and accuracy. As the availability of large-scale gameplay data grew, research shifted towards data-driven methods. **Yannakakis and Togelius (2013)** emphasized the importance of dynamic and adaptable player models, setting the stage for more sophisticated ML-based approaches. Additionally, **Drachen et al. (2009)** used clustering techniques to categorize player behavior in games, showing that understanding these patterns could be useful for predictive tasks, although the models at the time didn't directly focus on forecasting actions.

Supervised learning methods, such as classification and regression, have been popular for predicting player actions. For instance, **Suárez and Ontañón (2016)** applied decision trees and support vector machines (SVMs) to predict strategies in games

like StarCraft, demonstrating that ML could effectively anticipate player moves using game data. Similarly, **Andrade et al. (2005)** used Naive Bayes classifiers to predict actions in shooting games, showing that simpler ML models could be effective in identifying action patterns. However, as games became more complex, these traditional methods started to show their limitations in capturing the nuances of player behavior.

EXISTING SYSTEM

Existing systems use rule-based engines or hard- coded AI routines that do not generalize well across different players or games. These systems:

- Lack personalization
- Are not adaptive to player learning
- Do not improve with more gameplay data
- Struggle with sequential and time-based dependencies

Problem Statement

Current systems in gameplay analytics fail to predict complex human behavior dynamically. There's a need for a real-time prediction engine that can analyze game logs, player decisions, and state transitions to anticipate the next player move. This is vital in game AI development, e-sports strategy formulation, and behavior modeling.

Proposed System

The proposed system:

- Collects gameplay data including past actions, environmental states, and outcome feedback
- Cleans and preprocesses data using NLP and numeric encoding techniques
- Trains ML models (Random Forest, SVM, LSTM) to classify or predict the next action
- Uses sequential models for real-time action prediction
- Displays output through a dashboard for monitoring and AI decision support

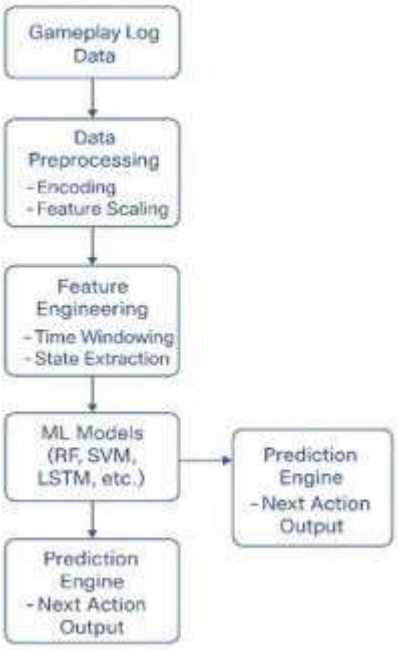
System Requirement Specification Functional Requirements

- Load and parse player activity/game logs
- Clean and preprocess input data
- Train and validate classification models
- Predict and visualize the next possible action
- Provide real-time feedback or data export

Architecture Overview

1. **Data Source**: Game logs, including actions, scores, player states, timestamps
2. **Preprocessing**: Converts raw log data into usable features, normalizes inputs, encodes labels
3. **Feature Engineering**: Derives behavioral patterns, timing intervals, and game states
4. **ML Model Training**: Uses classifiers (RF, SVM) or sequence models (LSTM) for prediction
5. **Prediction Engine**: Predicts the most probable next move based on current context
6. **Frontend/Dashboard**: Displays predictions for developers, analysts, or AI bots
7. **Feedback Loop**: Improves over time by integrating new gameplay data

Architecture Diagram



Implementation

The recommendation system was implemented using Python, leveraging its extensive libraries for data processing and machine learning. Initially, the dataset containing user preferences and item details was collected and thoroughly preprocessed by removing duplicates, handling missing values, and encoding categorical variables where necessary. Once the data was cleaned, feature extraction techniques were applied to identify meaningful patterns. For building the recommendation model, both content-based filtering and collaborative filtering approaches were explored. Content-based filtering analyzed the item attributes to suggest similar products, while collaborative filtering utilized user-item interaction data to predict preferences by identifying similarities between users or items. The model's performance was assessed using evaluation metrics such as Root Mean Square Error (RMSE) and Mean Absolute Error (MAE), ensuring its accuracy and reliability. Finally, the system was integrated into a simple user interface, allowing users to receive personalized and accurate recommendations based on their historical interactions and preferences.

MODULE DESCRIPTION

The **Player Action Prediction System** module is designed to leverage advanced machine learning techniques to predict player behavior within dynamic gaming environments. This system aims to analyze a player's previous actions, game state, and contextual factors to forecast the next action they will take. The primary objective is to enhance AI-driven decision-making processes in games, such as controlling non- player characters (NPCs), creating adaptive game experiences, or providing valuable insights for game developers to improve gameplay mechanics.

At its core, the system ingests a variety of data, including time-series sequences that record a player's movements, choices, and interactions with the game world. This data can encompass numerous features such as the player's current health, position on the map, proximity to other players or objects, in-game items, or even environmental conditions (e.g., weather, level of difficulty). By processing this information, the system identifies patterns and correlations that reflect a player's decision-making process.

Machine learning models, such as **Supervised Learning** algorithms (e.g., Decision Trees, Random Forests, Support Vector Machines), and **Deep Learning** methods (such as Recurrent Neural Networks (RNN) and Long Short-Term Memory (LSTM) networks), are employed to analyze these sequences. In the case of time-series data, the model learns from the past actions of the player to predict future actions, ensuring that the prediction accounts for both short-term and long-term

behavior. These models are trained using historical data that links the player's state to the actions they took at each point in time.

For **Supervised Learning**, the system treats each player action as a classification problem, where the task is to predict a specific action category (e.g., attack, jump, defend, or move in a certain direction). This approach requires labeled datasets, where the player's actions are annotated, providing the system with clear input-output pairs. On the other hand, **Reinforcement Learning** models can be used when the system operates in real-time, continuously learning and adapting through interactions with the game environment. This approach is well-suited for cases where a player's actions evolve based on rewards and punishments they receive from the game, such as improved performance or failure.

The module incorporates several key features, including **data preprocessing**, where the data is cleaned, normalized, and transformed for optimal model performance. Techniques like **feature engineering** are also applied to extract meaningful information from raw data (e.g., calculating distances, tracking item usage, or aggregating player behavior over time). Additionally, **hyperparameter tuning** and **cross-validation** are employed to refine model parameters and improve generalization.

Once trained, the model is evaluated using a variety of metrics to ensure it performs reliably and accurately. Common evaluation techniques include **accuracy**, **precision**, **recall**, and **F1 score**, especially when dealing with multi-class classification problems. In a reinforcement learning context, evaluation may also include assessing **cumulative rewards** or **policy effectiveness** based on the long-term performance of the agent (the player).

Finally, the module is deployed in real-time gaming systems, where it continuously predicts player actions based on current game states. It can be integrated into various game engines or used to control AI opponents, dynamically adjust game difficulty, or provide real-time feedback to enhance the player experience. Continuous **monitoring and retraining** are also essential to ensure that the model adapts to evolving player strategies or changes in game dynamics.

The **Player Action Prediction System** holds significant potential for game developers, as it can enable personalized player experiences, improve AI behavior in games, assist in balancing gameplay, and even provide insights for post-game analysis, such as understanding player behavior trends or detecting potential cheating. Ultimately, it enhances the realism and engagement of digital environments by making them more responsive to the individual player's choices.

CONCLUSION

In conclusion, this study presents a robust and explainable machine learning framework for predicting ICU length of stay (LOS) at the time of hospital admission. By integrating real EHR data classification algorithms and explainable AI (xAI) with these techniques, the system provides both high predictive accuracy and interpretability. The use of representations like XGBoost, combined with clear performance metrics and effective data preprocessing, ensures the framework is reliable and practical for clinical application.Most importantly, the explainability of the model outputs empowers healthcare professionals to make informed decisions based on transparent predictions. This methodology not only facilitates better hospital resource management but also enhances patient care by enabling timely interventions and planning. The framework marks a substantial breakthrough in applying artificial intelligence to healthcare, making predictive tools more accessible, trustworthy, and impactful in real-world hospital settings.

REFERENCES

1. A. Awad, M. Bader-El-Den, and J. McNicholas, "Patient length of stay and mortality prediction: A survey," *Health Services Manage. Res.*, vol. 30, no. 2, pp. 105–120, May 2017.

2. OECD.(2020). *Length of Hospital Stay (Indicator)*. Accessed: Jul. 21, 2021. [Online]. Available: https://data.oecd.org/healthcare/ length-of-hospital-stay.htm

3. Australian Institute of Health and *Public Hospitals in 2011–12*. [Online]. Available: https://www.aihw.gov.au/report s/hospitals/hospital-performance-length- ofstay- in-2011-12

4. F. Pecoraro, F. Clemente, and D. Luzi, "The efficiency in the ordinary hospital bed management in Italy: An in-depth analysis of intensive care unit in the areas affected by COVID-19 before the outbreak," *PLoS ONE*, vol. 15, no. 9, Sep. 2020

5. M. Hassan, H. P. Tuckman, R. H. Patrick, D. S. Kountz, and J. L. Kohn, "Hospital length of stay and probability of acquiring infection," Int. J. Pharm. Healthcare Marketing, vol. 4, no. 4, pp. 324–338, Nov. 2010.

6. M. C. Blom, K. Erwander, L. Gustafsson, M. Landin-Olsson, F. Jonsson, and K. Ivarsson, "The probability of readmission within 30 days of hospital discharge is positively associated with inpatient bed occupancy at discharge—A retrospective cohort study," BMC Emerg. Med., vol. 15, no. 1, pp. 1–6, Dec. 2015.

7.E. Rocheteau, P. Liò, and S. Hyland, "Temporal pointwise convolutional networks for length of stay prediction in the intensive care unit," arXiv preprint, arXiv:2007.09483, 2020.

8.C. W. Hanson, C. S. Deutschman, H. L. Anderson, P. M. Reilly, E. C. Behringer, C. W. Schwab, and J. Price, "Effects of an organized critical care service on outcomes and resource utilization: A cohort study," Crit. Care Med., vol. 27, no. 2, pp. 270–274, Feb. 1999.

9. S. Siddiqui, S. Ahmed, and R. Manasia, "Apache II score as a predictor of length of stay and outcome in our ICUs," J. Pakistan Med. Assoc., vol. 55, no. 6, p. 253, 2005.

10. W. A. Knaus, J. E. Zimmerman, D. P. Wagner, E. A. Draper, and D. E. Lawrence, "APACHE—Acute physiology and chronic health evaluation: A physiologically based classification system," Crit. Care Med., vol. 9, no. 8, pp. 591–597, Aug. 1981.