**DataPredict: A General Machine, Deep, Reinforcement Learning Library For Lua-Native Games**

**Aqwam Harish Aiman**

**Abstract**

Machine learning, deep learning and reinforcement learning frameworks such as scikit-learn, TensorFlow, PyTorch and Stable Baselines allow researchers to train models using real-world datasets. However, there is a noticeable gap for game-related frameworks. This has led to difficulties in producing game-related machine learning, deep learning and reinforcement learning research as researchers are forced to integrate frameworks whose programming languages are incompatible with the game engines. Aiming at mimicking scikit-learn design, DataPredict offers a way of implementing machine learning, deep learning and reinforcement learning into Lua-native game engines like Roblox Studio.

**Introduction**

Machine learning, deep learning and reinforcement learning usage have become increasingly common in general real-world applications. Many frameworks, like scikit-learn and PyTorch, speed up the implementation of these models for general real-world applications. However, this focus has led to neglect in simulated worlds that mimic real-world interactions, such as games.

The gaming industry is estimated to be worth over $184 billion. This presents a significant opportunity for commercialising such models on a large scale. In addition, gaming platforms like Roblox that support user-generated content provide a significant amount of data that could be used to train these models. Although model training on this user-generated content has been done, very few have taken advantage of game-generated data to build in-game models that adapt to players in real-time. If done correctly, the in-game models could allow games to retain players much longer, earn higher revenue and have higher player-returning-power.

**Model Selection Criteria**

Currently, DataPredict offers over 50 machine learning, deep learning and reinforcement learning models. This includes unsupervised learning, deep reinforcement learning and so on. However, choosing the most appropriate models that could be used in games is difficult. This is because games produce incomplete data in real-time and may never produce a full dataset. Additionally, games generally require large amounts of computational resources to simulate physics and graphics in games. Heavy use of these models could lead to a large decrease in game performance and negatively impact the players’ experience. As such, the models are generally selected based on these groups:

* Ability to perform online or incremental training.
* Sample efficiency
* Use cases
* Computational complexity