Player 1 vs AI

AI is a cutting-edge technology in the game industry. Deep learning as well as machine learning techniques have allowed games to become more intelligent. In current games, players are encouraged to think about their own patterns and then respond accordingly. AI is helping developers come up with new ways of doing things.

We've thought about the possibility of having two players, such as Player 1 and Player 2 above. However, there may be a situation when a person is both online and offline and has no one to play with. In order to play Stone, Paper, and Scissors with a computer, we've built an AI model of the game. In our search, we came across a variety of models as well as algorithms, from which we chose the most often used ones in games, which were:

* Firstly, the Markov chain model
* Second, SVM with multi-label classification
* Finally, RNN, along with LSTM and GRU

The neural network model was previously trained to classify the gesture as either stone, paper, or scissors before this. Here, we're using the model's current state to forecast its future state, or sequence. Among the most common examples of this sort of application is the weather forecast model, which includes parameters such as temperature, humidity, and climate change. In light of this information, the model forecasts what the weather will be like on the next day. However, in our model, the elements of the model—such as stones, paper, and scissors—serve as the model's characteristics. As a result, we may anticipate the model's future state/sequence using an existing sequence of stone, paper, and scissors.

# Markov Model:

A Markov chain is a mathematical concept that combines probability principles to move through one phase to another. Future probabilities are computed based on current values in a random process. The likelihood of a new state is influenced by the preceding condition or series. As a statistical technique for modelling and forecasting the impacts of future modifications, the Markov chain is used to calculate the likelihood of transfers among various states. Decisions made by humans are constantly influenced by their thoughts and feelings. In the game of stone-paper-scissors, this is also the case. Using this method, the outcome of the earlier games impacts the player's moves in the following game. The Markov matrix is a distinctive characteristic of such a clever system that incorporates the game outcomes into it.

Stone, paper, and scissors are the three states of play in the game. All three states are regulated by a total of 9 possibilities. Based on the previous move, the Markov model predicts the player's future move. For example, the previous sequence of the game is stored as Stone, Scissors, Scissors. Since the likelihood of stone occurring is larger than that of paper and scissors, the Markov model will verify the previous sequence of events when forecasting the player's next move and, based on that, predicts the next move as S, i.e., stone.

# Multi-label Classification Via SVM:

An SVM is a classification technique that uses supervised machine learning. The neural network model has previously categorised and classified the hand pattern as stone, paper, or scissors, and the SVM model utilises the user's past inputs to anticipate the next move. Once a user submits input, a for loop iteration is utilised to execute over the sequences. Repeatedly increasing the number of sequences alters the output. Stone, paper, and scissors are stored in a one-dimensional array, and this one-dimensional array is categorised into Multi-label (X and Y) after going through the sequences. For example, if the window size is 5 and the sequence is Stone, Stone, Stone, Paper, Paper, Scissors, Scissors, then the x would be Stone, Stone, Stone, Paper, Paper, and y, i.e., the prediction would be Scissors. The algorithm is used as a retrieval frame, in which it encodes the sequences and divides them into testing and training. As a result, it can predict what will happen next based on what has happened before.

# RNN, LSTM & GRU:

The observations are made using an RNN method that integrates LSTM and GRU. The sequential data has been analysed using recurrent neural networks. In Recurrent Neural Networks, there are loops, and they may be used to store and reuse data. Natural Language Processing (NLP) is one of the many tasks that require dealing with sequential data. This is possible thanks to RNNs. When it comes to sentence construction, we understand they may be as short or as long as we choose. A "many-to-one forecasting issue" occurs when numerous keywords are utilised to determine a single word. A feed-forward neural net is used to build a one-to-one RNN model, which is used in our model to predict the result. Starting with the neurons that receive input, it progresses to the hidden layer and the nodes are never reached twice. The RNN's principal job is to return data to the network. There are two inputs that may be used to estimate future moves: the current and the previous. Short-term memory is a strong suit of conventional RNNs, but as the distance between available details or where it is required widens, so does their effectiveness.

The LSTM theory is an enhancement of the RNN model that expands its memory. Inner gates are used to determine what data is preserved as well as what data is discarded in LSTMs. It makes it possible for new details to be entered and is relevant enough to have an effect on the final result. The concept of prioritising some data over others is a natural one since our brains tend to prioritise specific facts over others. To get the most out of your neural network, it's important to have a huge number of LSTM units. However, RNNs may be unwrapped into numerous linked units with the same architecture but varying weights in order to better understand how they work. In order to forecast the next play of the opponent, the LSTM model was constructed.

A Gated Recurrent Unit (GRU) is a type of RNN. Like LSTM, but with a smaller number of parameters, it is learned quicker. Hyper-parameters such as input size and batch size, layers and classes are all configured in the RNN model at the start of the training process. A function is developed and constraints are added in order to evaluate movements. The model is constructed, and the RNN model is fed a series of inputs. Additional movements may be predicted based on the user's prior actions using an RNN model.