Name : Shruti Chidrawar (161)
Vasavi kawtikwar (166)
Project name : Rock-Paper-Scissor Predictor

Report

Aim

The aim of this project is to design and implement an AI-based Rock-Paper-Scissor (RPS) Predictor, which predicts the user's next move (Rock, Paper, or Scissor) based on historical pattern analysis. The objective is to build an intelligent system that learns from user behavior and makes accurate predictions using Artificial Intelligence techniques such as Machine Learning classification, probability modeling, or pattern recognition.

This project demonstrates how AI can be applied to decision-making, pattern learning, and predictive modeling.

Stack

Programming Language: Python
Concept Used: Artificial Intelligence, Machine Learning (Classification & Pattern Recognition)
Paradigm: Predictive Modeling and Statistical Learning
Key Features

- Data collection of user moves (Rock, Paper, Scissor)
- Learning user patterns using probability or ML model
- Real-time prediction of user's next move
- Automatic computer response selection to try to win
- Continuous model updating with each new move

Software Requirements

- Python (3.x)
- Required libraries: NumPy, Pandas, Scikit-learn (if ML model used)
- Jupyter Notebook / PyCharm / VS Code

Hardware Requirements

- Standard computer system capable of running Python and ML computations

Algorithm

1. Initialize Game / Dataset
    - Start by initializing counters for Rock, Paper, and Scissor.
    - If a dataset is used, load previous game history.
    - Set variables for storing user's previous moves.
2. Collect User Input
    - User selects Rock, Paper, or Scissor.
    - Store every move for pattern learning.
3. Pattern Analysis / Prediction
    - Apply one or more prediction strategies:
    a. Frequency-based prediction
    The AI predicts the move the user plays most often.
    b. Markov chain pattern detection
    AI predicts the next move based on transition probability.
    c. Machine Learning classifier
    (Optional) Train a model like Logistic Regression, Decision Tree, or Naive Bayes.
4. Computer Move Generation
    - Based on predicted user move:
    a. If user is predicted to play Rock → computer plays Paper
    b. If user is predicted to play Paper → computer plays Scissor
    c. If user is predicted to play Scissor → computer plays Rock
5. Determine Winner
    - Compare predicted computer move with user’s actual move.
    - Update scores and dataset.
6. Continuous Learning
    - After each round:
    a. Update the model with new inputs.
    b. Improve prediction accuracy.
7. Game End / Exit
    - User stops the game.

- Display accuracy and summary of predictions.

Steps of Working or Working Mechanism

1. Start the Program
   Run the Python script to initialize the model and start the game.
2. User Enters Move
   The user selects Rock, Paper, or Scissor.
3. Prediction Process
   - System reads past inputs.
   - AI model predicts the next user move.
4. Computer Chooses Counter Move
   The system selects the winning move against the predicted one.
5. Result Display
   - Shows who won each round.
   - Displays prediction accuracy.
6. Model Update
   The system updates its learning model using each new move.
7. End of Game
   When the user ends the game, the system displays:
   - Accuracy
   - Total rounds played
   - User move distribution
   - Prediction performance

Conclusion

The Rock-Paper-Scissor Predictor successfully demonstrates Artificial Intelligence concepts through predictive modeling and pattern analysis. By collecting user inputs and learning from their historical behavior, the system intelligently predicts future moves and adapts its strategy over time. This project highlights how machine learning and probability-based reasoning can be applied in game environments to build intelligent agents capable of analyzing patterns, making predictions, and improving performance with continuous interactions.