AI Smart Tic-Tac-Toe: Minimax SVM 🤖

In this project, I developed an AI-powered Tic-Tac-Toe game using the Support Vector Machine (SVM) algorithm alongside the minimax strategy for optimal gameplay. The project involved several critical steps to ensure the AI could play against a human player effectively.

1. Loading and Reading Dataset Header 📂

I initiated the project by loading the dataset containing Tic-Tac-Toe game results using Pandas. This involved reading a CSV file that captures various game moves and their outcomes, which laid the foundation for further analysis.

2. Handling Missing Values 🚫

To ensure data integrity, I replaced any missing values (represented as '?') with a placeholder (9) to indicate an invalid move. This step was crucial in preventing errors during analysis and modeling, and I confirmed the absence of missing values post-processing.

3. Data Preparation and Preprocessing ⚙️

I mapped the target variable ('CLASS') to numerical values (0 for loss, 1 for win, 2 for draw), which is essential for training machine learning models. Additionally, I converted the move columns to integers for consistency and better compatibility with the SVM algorithm.

4. Exploratory Data Analysis (EDA) 📊

Using Seaborn and Matplotlib, I conducted EDA to visualize the distribution of game outcomes and the correlation between features. The countplot revealed the frequency of each outcome, while the correlation matrix highlighted relationships among the move columns, aiding in feature selection.

5. Splitting Data, Validation, and Cross-Validation 🔍

I split the dataset into features (X) and target (y), followed by creating training and testing sets (70% training, 30% testing). Cross-validation was employed to evaluate model performance, resulting in scores that indicated consistent prediction accuracy across different data subsets.

6. Hyperparameter Tuning 🛠️

To enhance model performance, I implemented Grid Search to identify optimal hyperparameters for the SVM model. The grid search evaluated multiple configurations, leading to the selection of parameters that maximized model accuracy during training.

7. Modeling with SVM 🧠

With the best hyperparameters identified, I trained the SVM model. The choice of SVM was driven by its effectiveness in classification tasks, allowing the model to learn complex patterns in the game data.

8. Evaluation 📈

I assessed the model's performance using the testing set, yielding an accuracy of approximately **X%** (insert actual measure). The classification report provided insights into precision, recall, and F1 scores, indicating how well the model distinguished between game outcomes.

9. Game Interface Using GUI 🎮

I developed a user-friendly graphical interface using Tkinter, allowing users to interact with the AI in real-time. The minimax algorithm was integrated to enable the AI to make optimal moves, significantly enhancing the gameplay experience.

Results and Interpretation

The project successfully demonstrated the AI's capability to play Tic-Tac-Toe at a high level, learning from historical game data. The accuracy achieved indicates that the model can predict game outcomes effectively, providing a robust foundation for future enhancements or additional features.

Skills Acquired:

Data Preprocessing, Exploratory Data Analysis, Machine Learning, Hyperparameter Tuning, SVM, Model Evaluation, GUI Development, Minimax Algorithm.

Hashtags:

#DataScience #MachineLearning #AI #TicTacToe #SVM #DataAnalysis #Python #Pandas #Seaborn #Matplotlib #HyperparameterTuning #GridSearch #ExploratoryDataAnalysis #GameDevelopment #ArtificialIntelligence #DeepLearning #DataVisualization #SoftwareEngineering #ComputerScience #Technology