Tic-Tac-Toe AI Using Minimax Algorithm

Tic-Tac-Toe is a classic two-player game played on a 3x3 grid.

This project implements an unbeatable AI using the Minimax algorithm.

The player uses 'X' and the AI uses 'O'. The AI always plays optimally.

Python Code:

```
import math

def print_board(board):
    print("\n")
    for row in board:
        print("|".join(row))
        print("-" * 5)

def check_winner(board):
    win_states = [
        [board[0][0], board[0][1], board[0][2]],
        [board[1][0], board[1][1], board[1][2]],
        [board[2][0], board[2][1], board[2][2]],
        [board[0][0], board[1][0], board[2][0]],
        [board[0][1], board[1][1], board[2][1]],
        [board[0][2], board[1][2], board[2][2]],
        [board[0][0], board[1][1], board[2][2]],
        [board[0][2], board[1][1], board[2][0]],
    ]
    for line in win_states:
        if line.count('X') == 3:
            return 'X'
        if line.count('O') == 3:
            return 'O'
    return None

def is_draw(board):
    for row in board:
        if ' ' in row:
            return False
    return True

def minimax(board, depth, is_maximizing):
    winner = check_winner(board)
    if winner == 'O':
        return 1
    elif winner == 'X':
        return -1
    elif is_draw(board):
        return 0

    if is_maximizing:
        best_score = -math.inf
        for i in range(3):
            for j in range(3):
                if board[i][j] == ' ':
                    board[i][j] = 'O'
```

```
                    score = minimax(board, depth + 1, False)
                    board[i][j] = ' '
                    best_score = max(score, best_score)
        return best_score
    else:
        best_score = math.inf
        for i in range(3):
            for j in range(3):
                if board[i][j] == ' ':
                    board[i][j] = 'X'
                    score = minimax(board, depth + 1, True)
                    board[i][j] = ' '
                    best_score = min(score, best_score)
        return best_score

def ai_move(board):
    best_score = -math.inf
    move = None
    for i in range(3):
        for j in range(3):
            if board[i][j] == ' ':
                board[i][j] = 'O'
                score = minimax(board, 0, False)
                board[i][j] = ' '
                if score > best_score:
                    best_score = score
                    move = (i, j)
    return move

def play_game():
    board = [[' ' for _ in range(3)] for _ in range(3)]
    print("Welcome to Tic-Tac-Toe!")
    print("You are X, AI is O.")
    print_board(board)

    while True:
        while True:
            try:
                row = int(input("Enter row (0, 1, 2): "))
                col = int(input("Enter col (0, 1, 2): "))
                if board[row][col] == ' ':
                    board[row][col] = 'X'
                    break
                else:
                    print("Cell already taken.")
            except (IndexError, ValueError):
                print("Invalid input. Try again.")

        print_board(board)

        if check_winner(board):
            print("You win!" if check_winner(board) == 'X' else "AI wins!")
            break
        if is_draw(board):
            print("It's a draw!")
            break

        ai_r, ai_c = ai_move(board)
        board[ai_r][ai_c] = 'O'
        print("\nAI has moved:")
        print_board(board)
```

```
        if check_winner(board):
            print("AI wins!" if check_winner(board) == 'O' else "You win!")
            break
        if is_draw(board):
            print("It's a draw!")
            break

if __name__ == "__main__":
    play_game()
```

Sample Output:

```
Welcome to Tic-Tac-Toe!
You are X, AI is O.

 | |
-----
 | |
-----
 | |
Enter row (0, 1, 2): 0
Enter col (0, 1, 2): 0

X| |
-----
 | |
-----
 | |

AI has moved:
X| |
-----
 |O|
-----
 | |

... game continues ...
```