PRACTICAL 1

Topic: TicTacToe solver

Introduction: I have made an ai bades tictac toe player which works on minmax algo, minimizing the opponent and maximizing itself.

Code:

import math

class TicTacToe:

    def \_\_init\_\_(self):

        # Initialize a 3x3 board with empty spaces

        self.board = [[" " for \_ in range(3)] for \_ in range(3)]

    def display\_board(self):

        """Display the current state of the board."""

        for row in self.board:

            print("|".join(row))

            print("-" \* 5)

    def is\_winner(self, player):

        """Check if the given player has won."""

        # Check rows, columns, and diagonals

        for row in self.board:

            if all(cell == player for cell in row):

                return True

        for col in range(3):

            if all(self.board[row][col] == player for row in range(3)):

                return True

        if all(self.board[i][i] == player for i in range(3)) or all(

            self.board[i][2 - i] == player for i in range(3)

        ):

            return True

        return False

    def is\_full(self):

        """Check if the board is full."""

        return all(cell != " " for row in self.board for cell in row)

    def minimax(self, is\_maximizing):

        """Minimax algorithm to find the best move."""

        # Check for terminal states

        if self.is\_winner("X"):

            return 1  # AI wins

        if self.is\_winner("O"):

            return -1  # Opponent wins

        if self.is\_full():

            return 0  # Draw

        # Maximizing player (AI - 'X')

        if is\_maximizing:

            best\_score = -math.inf

            for i in range(3):

                for j in range(3):

                    if self.board[i][j] == " ":

                        self.board[i][j] = "X"

                        score = self.minimax(False)

                        self.board[i][j] = " "

                        best\_score = max(best\_score, score)

            return best\_score

        # Minimizing player (Opponent - 'O')

        else:

            best\_score = math.inf

            for i in range(3):

                for j in range(3):

                    if self.board[i][j] == " ":

                        self.board[i][j] = "O"

                        score = self.minimax(True)

                        self.board[i][j] = " "

                        best\_score = min(best\_score, score)

            return best\_score

    def best\_move(self):

        """Find the best move for the AI."""

        best\_score = -math.inf

        move = None

        for i in range(3):

            for j in range(3):

                if self.board[i][j] == " ":

                    self.board[i][j] = "X"  # Simulate AI move

                    score = self.minimax(False)

                    self.board[i][j] = " "  # Undo move

                    if score > best\_score:

                        best\_score = score

                        move = (i, j)

        return move

    def play(self):

        """Play a game of Tic-Tac-Toe."""

        print("Welcome to Tic-Tac-Toe!")

        print("You are 'O', AI is 'X'.")

        print("Enter your move as 'row col' (e.g., 1 2).")

        self.display\_board()

        while not self.is\_full():

            # Player's turn

            row, col = map(int, input("Your move (row col): ").split())

            if self.board[row][col] == " ":

                self.board[row][col] = "O"

            else:

                print("Invalid move! Try again.")

                continue

            self.display\_board()

            if self.is\_winner("O"):

                print("You win!")

                return

            if self.is\_full():

                break

            # AI's turn

            print("AI's turn...")

            move = self.best\_move()

            if move:

                self.board[move[0]][move[1]] = "X"

            self.display\_board()

            if self.is\_winner("X"):

                print("AI wins!")

                return

        print("It's a draw!")

# Play the game

game = TicTacToe()

game.play()

Output:

