Lab-1

Date 04/10/2024
Page

☆ Tic-Tac-Toe implementation using Python

→ Pseudocode

```
Function minimax(node, depth, isMaximizingPlayer):
    if node is a terminal state:
        return evaluate(node)

    if isMaximizingPlayer:
        bestValue = -inf
        for each child in node:
            value = minimax(child, depth, false)
            bestValue = max(bestValue, value)
        return bestValue

    else:
        bestValue = +inf
        for each child in node:
            value = minimax(child, depth, true)
            bestValue = min(bestValue, value)
        return bestValue
```

tictactoe

November 9, 2024

[3]:
```python
# TICTACTOE
print("Name: Sudarshan Komar", "USN: 1BM22CS291", sep="\n")

board = {1: ' ', 2: ' ', 3: ' ',
         4: ' ', 5: ' ', 6: ' ',
         7: ' ', 8: ' ', 9: ' '}

def printBoard(board):
    print(board[1] + '|' + board[2] + '|' + board[3])
    print('-+-+-')
    print(board[4] + '|' + board[5] + '|' + board[6])
    print('-+-+-')
    print(board[7] + '|' + board[8] + '|' + board[9])
    print('\n')

def spaceFree(pos):
    return board[pos] == ' '

def checkWin():
    win_conditions = [(1, 2, 3), (4, 5, 6), (7, 8, 9), (1, 4, 7),
                      (2, 5, 8), (3, 6, 9), (1, 5, 9), (3, 5, 7)]
    for x, y, z in win_conditions:
        if board[x] == board[y] == board[z] != ' ':
            return True
    return False

def checkDraw():
    return all(board[key] != ' ' for key in board)

def insertLetter(letter, position):
    if spaceFree(position):
        board[position] = letter
        printBoard(board)

        if checkWin():
            print(f'{letter} wins!')
            return True
```

```
        elif checkDraw():
            print('Draw!')
            return True
    else:
        print('Position taken, please pick a different position.')
    return False

player = 'O'
bot = 'X'

def playerMove():
    position = int(input('Enter position for O: '))
    while not spaceFree(position):
        position = int(input('Position taken. Enter a new position for O: '))
    return insertLetter(player, position)

def compMove():
    bestScore = -1000
    bestMove = 0
    for key in board.keys():
        if spaceFree(key):
            board[key] = bot
            score = minimax(board, False)
            board[key] = ' '
            if score > bestScore:
                bestScore = score
                bestMove = key
    return insertLetter(bot, bestMove)

def minimax(board, isMaximizing):
    if checkWin():
        return 1 if not isMaximizing else -1
    elif checkDraw():
        return 0

    if isMaximizing:
        bestScore = -1000
        for key in board.keys():
            if spaceFree(key):
                board[key] = bot
                score = minimax(board, False)
                board[key] = ' '
                bestScore = max(score, bestScore)
        return bestScore
    else:
        bestScore = 1000
        for key in board.keys():
```

```
            if spaceFree(key):
                board[key] = player
                score = minimax(board, True)
                board[key] = ' '
                bestScore = min(score, bestScore)
        return bestScore

printBoard(board)
gameOver = False
while not gameOver:
    gameOver = compMove() or playerMove()
```

```
Name: Sudarshan Komar
USN: 1BM22CS291
 | |
-+-+-
 | |
-+-+-
 | |


X| |
-+-+-
 | |
-+-+-
 | |


Enter position for O: 5
X| |
-+-+-
 |O|
-+-+-
 | |


X|X|
-+-+-
 |O|
-+-+-
 | |


Enter position for O: 3
X|X|O
-+-+-
 |O|
```

```
-+-+-
 | |


X|X|O
-+-+-
 |O|
-+-+-
X| |


Enter position for O: 4
X|X|O
-+-+-
O|O|
-+-+-
X| |


X|X|O
-+-+-
O|O|X
-+-+-
X| |


Enter position for O: 8
X|X|O
-+-+-
O|O|X
-+-+-
X|O|


X|X|O
-+-+-
O|O|X
-+-+-
X|O|X


Draw!
```