**28.** Write a program in python to perform Reinforcement Learning where an agent is trying to come out of a maze. It can move one random square or area in any direction, and get a reward if exits. The most common way to formalize a reinforcement problem is to represent it as Markov decision process. Assume the agent is in state b (maze area) and the target is to reach state f. So within one step agent can reach from b to f, let’s put a reward of 100 (otherwise 0) for links between nodes that allows agents to reach target state.

Consider the following matrix as the maze:

[[-1, -1, -1, -1, 0, -1],

[-1, -1, -1, 0, -1, 100],

[-1, -1, -1, 0, -1, -1],

[-1, 0, 0, -1, 0, -1],

[ 0, -1, -1, 0, -1, 100],

[-1, 0, -1, -1, 0, 100]]

**29.** Write a program in python to implement Upper Confidence Bound-1 algorithm on Ads\_Optimisation.csv data. Display the initial normalized values, the total score and final normalized values for each ad.

**30.** Write a program in python to implement Thompson Sampling algorithm for solving the K-armed Bandit Problem. Consider 5 slot machine (B1, B2, B3, B4 and B5) and generate 200 random observations for each slot machine. The 0’s to represent penalties or the player not getting a reward and all the 1’s to represent the player winning a reward while pulling the arm of the slot machine. Display the rewards per machine, total rewards and Machine Selected At Each Round By Thompson Sampling. Provide a histogram plot for displaying the rewards from each slot machine.

**31.** Write a program in python to implement Xgboost technique on the Diabetes dataset. Display the training and testing results using Scikit Learn.

**32.** Write a program in python to implement the Stratified K-fold Cross validation technique on the Diabetes dataset using n=5. Use Scikit Learn and display the ROC curves using Pyplot form Matplotlib.