Report - 1
on
Artificial Intelligence Lab Assignments

Adit Jain 201851007
IIIT Vadodara
Computer Science and Engineering
Email: 201851007@iiitvadodara.ac.in

Deep Shah 201851037
IIIT Vadodara
Computer Science and Engineering
Email: 201851037@iiitvadodara.ac.in

Devansh Agarwal 201851038
IIIT Vadodara
Computer Science and Engineering
Email: 201851038@iiitvadodara.ac.in

Kartikay Sarswat 201851057
IIIT Vadodara
Computer Science and Engineering
Email: 201851057@iiitvadodara.ac.in

Pallavi Sharma 201851079
IIIT Vadodara
Computer Science and Engineering
Email: 201851079@iiitvadodara.ac.in

Contents

***Abstract*—Taking Artificial Intelligence to a step ahead, in this assignment we have implemented Bayesian Networks, implemented classification, Learned and Implemented Markov models, Used expectation maximization algorithm, Explored e-greedy algorithms and q-learning using n-armed bandits and understood the process of sequential decision making. The scope of artificial intelligence is increased even further with the help of Q-Learning Algorithms. These implementations and algorithms prove how powerful is Artificial Intelligence and shows that its abilities are limitless.**

1. Introduction

In the past, the phenomenons that made life more convenient for our ancestors were termed to be “magic” and “supernatural powers”. Now? It’s called technology. And the “magic spells” are now termed as the complicated algorithms that get this technology running. One such technology that has occupied a prominent place in our present lives is Artificial Intelligence.

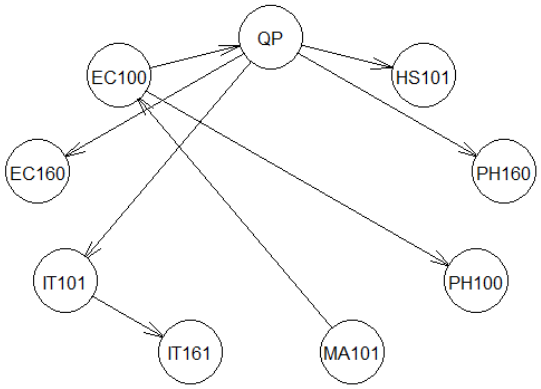
April 27, 2021

2. Assignment 5

2.1. Solution for Part A

Dependencies between all the courses,

```
Bayesian network learned via Score-based methods

model:
 [MA101][EC100|MA101][PH100|EC100][QP|EC100][EC160|QP][IT101|QP][PH160|QP]
 [HS101|QP][IT161|IT101]
nodes:                                9
arcs:                                 8
  undirected arcs:                    0
  directed arcs:                      8
average markov blanket size:          1.78
average neighbourhood size:           1.78
average branching factor:             0.89
```



2.2. Solution for Part B

Using data as provided in the txt file and on the basis of above dependencies, learn about the conditional probabilities for each course node through conditional probability table,

Starting from CPT of MA101/EC100 and then so on:-

```
Console  Terminal  Jobs
~/

  Bayesian network parameters

  Parameters of node EC100 (multinomial distribution)

Conditional probability table:

     MA101
EC100          AA          AB          BB          BC          CC          CD
   AA 0.75000000 0.07692308 0.03846154 0.01851852 0.00000000 0.00000000
   AB 0.00000000 0.46153846 0.25000000 0.05555556 0.00000000 0.00000000
   BB 0.25000000 0.23076923 0.32692308 0.22222222 0.04081633 0.00000000
   BC 0.00000000 0.15384615 0.28846154 0.27777778 0.32653061 0.00000000
   CC 0.00000000 0.07692308 0.09615385 0.24074074 0.32653061 0.04166667
   CD 0.00000000 0.00000000 0.00000000 0.12962963 0.26530612 0.33333333
   DD 0.00000000 0.00000000 0.00000000 0.03703704 0.04081633 0.50000000
   F  0.00000000 0.00000000 0.00000000 0.01851852 0.00000000 0.12500000
     MA101
EC100          DD          F
   AA 0.00000000 0.00000000
   AB 0.00000000 0.00000000
   BB 0.00000000 0.00000000
   BC 0.00000000 0.00000000
   CC 0.00000000 0.00000000
   CD 0.04761905 0.00000000
   DD 0.19047619 0.00000000
   F  0.76190476 1.00000000

  Parameters of node EC160 (multinomial distribution)

Conditional probability table:

     QP
EC160          n          y
   AA 0.00000000 0.07500000
   AB 0.00000000 0.10000000
   BB 0.01388889 0.18750000
   BC 0.01388889 0.36250000
   CC 0.15277778 0.22500000
   CD 0.44444444 0.03125000
   DD 0.26388889 0.01875000
   F  0.11111111 0.00000000
```

```
  Parameters of node IT101 (multinomial distribution)

Conditional probability table:

     QP
IT101          n          y
   AA 0.00000000 0.07500000
   AB 0.00000000 0.15625000
   BB 0.04166667 0.19375000
   BC 0.02777778 0.29375000
   CC 0.13888889 0.20000000
   CD 0.30555556 0.08125000
   DD 0.31944444 0.00000000
   F  0.16666667 0.00000000

  Parameters of node IT161 (multinomial distribution)

Conditional probability table:

     IT101
IT161          AA          AB          BB          BC          CC          CD
   AA 0.58333333 0.24000000 0.14705882 0.04081633 0.00000000 0.00000000
   AB 0.16666667 0.40000000 0.29411765 0.02040816 0.04761905 0.00000000
   BB 0.16666667 0.24000000 0.32352941 0.20408163 0.11904762 0.02857143
   BC 0.08333333 0.04000000 0.20588235 0.36734694 0.38095238 0.17142857
   CC 0.00000000 0.04000000 0.00000000 0.24489796 0.33333333 0.31428571
   CD 0.00000000 0.04000000 0.02941176 0.10204082 0.09523810 0.31428571
   DD 0.00000000 0.00000000 0.00000000 0.02040816 0.02380952 0.14285714
   F  0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.02857143
     IT101
IT161          DD          F
   AA 0.00000000 0.00000000
   AB 0.00000000 0.00000000
   BB 0.00000000 0.00000000
   BC 0.00000000 0.00000000
   CC 0.08695652 0.16666667
   CD 0.52173913 0.08333333
   DD 0.39130435 0.58333333
   F  0.00000000 0.16666667

  Parameters of node MA101 (multinomial distribution)

Conditional probability table:
          AA          AB          BB          BC          CC          CD          DD
0.01724138 0.05603448 0.22413793 0.23275862 0.21120690 0.10344828 0.09051724
           F
0.06465517
```

```
  Parameters of node PH100 (multinomial distribution)

Conditional probability table:

     EC100
PH100         AA          AB          BB          BC          CC          CD
   AA 0.71428571 0.40909091 0.22857143 0.08333333 0.00000000 0.00000000
   AB 0.14285714 0.31818182 0.20000000 0.18750000 0.05555556 0.00000000
   BB 0.00000000 0.18181818 0.31428571 0.29166667 0.13888889 0.03448276
   BC 0.14285714 0.04545455 0.14285714 0.22916667 0.33333333 0.13793103
   CC 0.00000000 0.04545455 0.11428571 0.18750000 0.25000000 0.41379310
   CD 0.00000000 0.00000000 0.00000000 0.02083333 0.19444444 0.31034483
   DD 0.00000000 0.00000000 0.00000000 0.00000000 0.02777778 0.10344828
   F  0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
     EC100
PH100         DD          F
   AA 0.00000000 0.00000000
   AB 0.00000000 0.00000000
   BB 0.05000000 0.00000000
   BC 0.00000000 0.00000000
   CC 0.20000000 0.02857143
   CD 0.45000000 0.11428571
   DD 0.20000000 0.45714286
   F  0.10000000 0.40000000

  Parameters of node PH160 (multinomial distribution)

Conditional probability table:

     QP
PH160          n          y
   AA 0.05555556 0.14375000
   AB 0.09722222 0.15625000
   BB 0.02777778 0.17500000
   BC 0.18055556 0.34375000
   CC 0.29166667 0.13750000
   CD 0.19444444 0.04375000
   DD 0.12500000 0.00000000
   F  0.02777778 0.00000000

  Parameters of node HS101 (multinomial distribution)

Conditional probability table:

     QP
HS101          n          y
   AA 0.00000000 0.26250000
   AB 0.00000000 0.21250000
   BB 0.05555556 0.22500000
   BC 0.12500000 0.16875000
   CC 0.18055556 0.08125000
   CD 0.19444444 0.03750000
   DD 0.37500000 0.01250000
   F  0.06944444 0.00000000
```

```
  Parameters of node QP (multinomial distribution)

Conditional probability table:

   EC100
QP         AA        AB        BB        BC        CC        CD        DD
  n 0.0000000 0.0000000 0.0000000 0.0000000 0.1388889 0.4482759 0.9500000
  y 1.0000000 1.0000000 1.0000000 1.0000000 0.8611111 0.5517241 0.0500000
   EC100
QP          F
  n 1.0000000
  y 0.0000000
```

2.3. Solution for Part C

On the basis of provided result i.e. student earns DD in EC100, CC in IT101 and CD in MA101, we have to find out grade in PH100

By using probability and conditional probability table as done in above case we get,

[1] "The grade student will get in PH100 is CD "

with **max probability of 0.4098.**

2.4. Solution for Part D

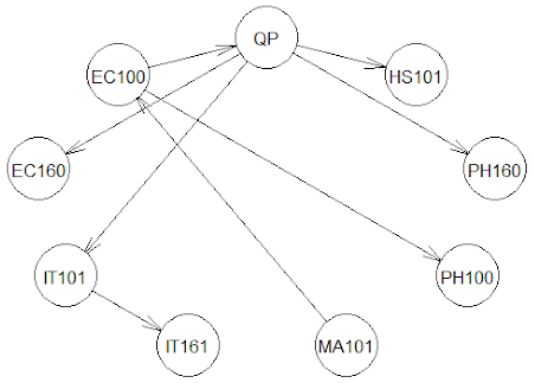
- Here, we are taking 70 percent data for training and building a naive Bayes Classifier and testing classifier on the remaining 30 percent data,
  Also , consider that the grades earned in different courses are independent of each other.
  Results:
  Trained Accuracy
  [1,] 0.9935
  Test Accuracy
  [1,] 0.974
- Repeating above case for 20 random selection of training and testing data,
  Results:
  Trained Accuracy
  [1,] 1
  Test Accuracy
  [1,] 0.8571

2.5. Solution for Part E

Consider that the grades earned in different courses may be dependent, which means that we will also have dependencies in between the courses.

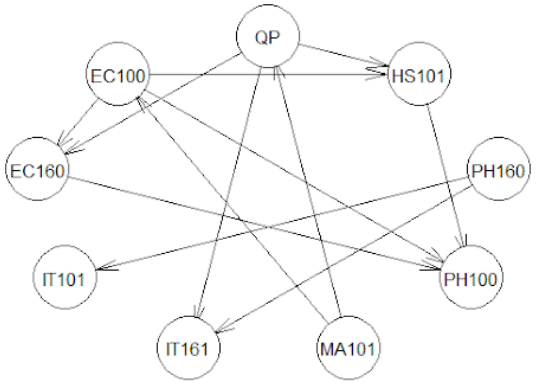
- Repeat question-4 Part-A,

```
> print(round(cbind("Trained Accuracy" =accuracy_tr), 4))
     Trained Accuracy
[1,]           0.9221
> print(round(cbind("Test Accuracy" =accuracy_test), 4))
     Test Accuracy
[1,]        0.8718
```



- Repeat question-4 Part-B,

```
> plot(tr.hc)
> print(round(cbind("Trained Accuracy" =accuracy_tr), 4))
     Trained Accuracy
[1,]           0.9167
> print(round(cbind("Test Accuracy" =accuracy_test), 4))
     Test Accuracy
[1,]        0.8571
>
```



Observation: This assignment deals with implementation and training of Bayesian Network.

3. Assignment 6

3.1. Objective

The objective of this assignment is to learn and implement EM optimization routine for learning parameters of hmm, to be able to use EM framework for deriving algorithms for problems with hidden or partial information.

3.2. Solution for Part A

This are the values obtained by training over war and peace by Leo Tolstoy by taking 1500 characters from the book.then the HMM layers are trained until minimum values are obtained. The HMM implementation contains alpha beta and gamma pass.

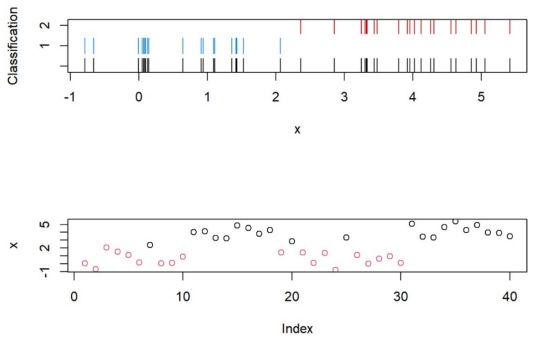
3.3. Solution for Part B

In this question we try to maximize the theta parameter assumed for a incomplete dataset and try to find theta such that the log probability logP(x;) of the observed data is maximised. given below are the output for 10 bent coins tossed 100 times.

```
Initial Theta
   0.5226   0.5911   0.1414   0.6579   0.7088   0.4528   0.9770   0.4377   0.7161   0.6568
New Theta
final =

 Columns 1 through 8:

   0.287560   0.391205   0.010000   0.569791   0.688256   0.176789   0.897061   0.092424

 Columns 9 and 10:

   0.784089   0.477410

 Columns 1 through 8:

   0.287560   0.391205   0.010000   0.569791   0.688256   0.176789   0.897061   0.092424

 Columns 9 and 10:

   0.784089   0.477410
```

3.4. Solution for Part C

The first graph obtained in the output is of using mcluster which is a clusterising algorithm using expectation maximization. The second graph is obtained by using kmeans which is a specific type of clustering algorithm of EM maximization routine on guassian variable.



4. Assignment 9

Multi armed bandits :-

- The multi-armed bandit problem is used in reinforcement learning to formalize the notion of decision-making under uncertainty. In a multi-armed bandit problem, an agent(learner) chooses between k different actions and receives a reward based on the chosen action.
  The multi-armed bandits are also used to describe fundamental concepts in reinforcement learning, such as rewards, timesteps, and values.
- For selecting an action by an agent, we assume that each action has a separate distribution of rewards and there is at least one action that generates maximum numerical reward. Thus, the probability distribution of the rewards corresponding to each action is different and is unknown to the agent(decision-maker). Hence, the goal of the agent is to identify which action to choose to get the maximum reward after a given set of trials.

4.1. Solution for Part A

Consider a binary bandit with two rewards 1-success, 0-failure. The bandit returns 1 or 0 for the action that you select, i.e. 1 or 2. The rewards are stochastic (but stationary). Use epsilon-greedy algorithm discussed in class and decide upon the action to take for maximizing the expected reward. There are two binary bandits named binaryBanditA.m and binaryBanditB.m are waiting for you.

Epsilon Greedy Algorithm :- The e-greedy algorithm shows us the importance of explorations vs exploitation.
Exploration :-It refers to exploring the environment randomly in order to gain more knowledge and to apply exploitation more precisely.
Exploration :-It refers to using the current knowledge to maximize the rewards.
Algo:-

```
If rand < e:
        Exploration
Else:
        Exploitation
```

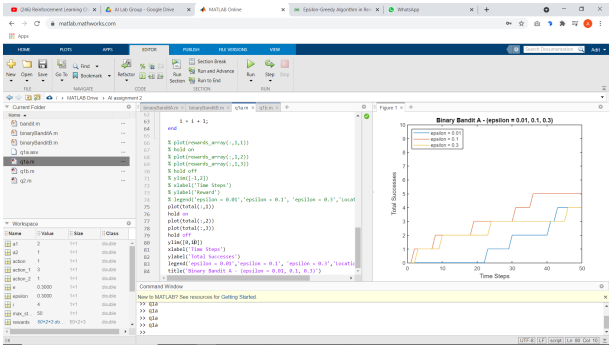
Hence both cannot be used simultaneously, but are equally important.

Binary bandits :-

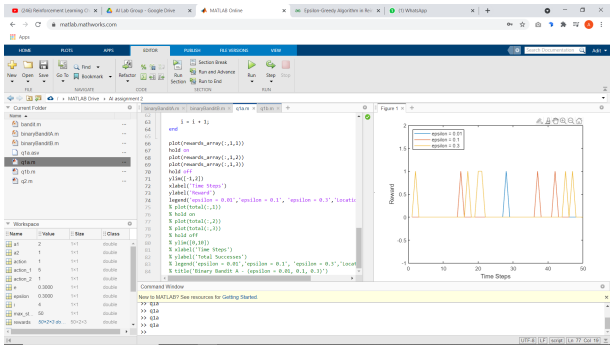
- Multi armed bandits with k = 2 are binary bandits. They have two rewards i.e. 1-success, 0-failure. The bandit returns 1 or 0 for the action that we select, i.e. 1 or 2. Rewards are stochastic (but stationary).
- Our epsilon ranges are [0.01,0.1,0.3]
- We have taken the number of time steps as 50.

In the first part we have probabilities as :-
p = [.1 .2]
So we have a low chance of getting success.

The following images show :-
difference in total number of successes for the range of epsilons
The rewards for the range of epsilons.

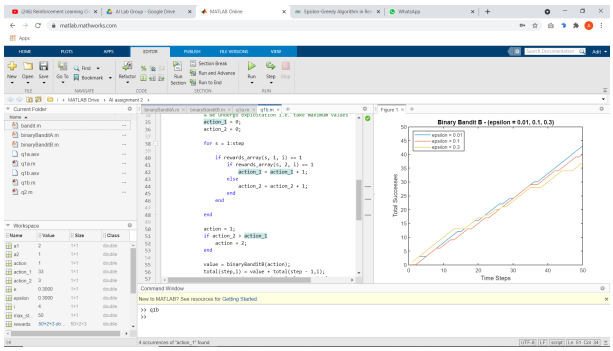


- Total number of successes vs Time steps compared by epsilons



- Rewards vs time steps compared by epsilons.

Similiarly, for the second part we have :-
P= [0.8,0.9]
So we have great chances of successes. The following figures show so :-

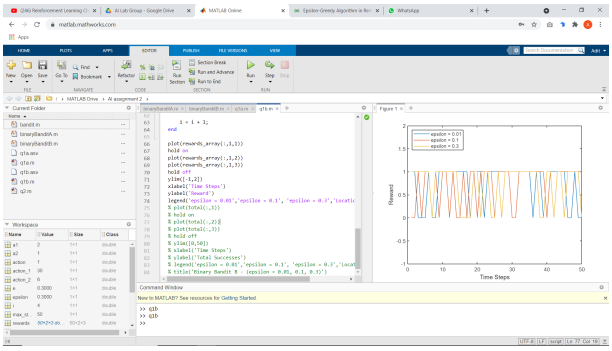


Ans: Here we have ten rewards, all non stationary. So they start as equals, but then change over time by adding a normally distributed increment with mean zero and standard deviation 0.01 to all mean-rewards on each time step.

- Value of epsilon used = 0.1
- Time steps = 10,000
- Mean rewards are [.1.2.25.3.35.4.5.7.45.55] and they are normally disributed as: reward = m(action) + randn.

- Total number of successes vs Time steps compared by epsilons

The figures below show:-



- Action Taken on a time step
- The number of times a particular action is taken
- Total successes with respect to time steps.
- Total value earned after a time step
- Average reward received with respect to time steps.

- Rewards vs time steps compared by epsilons.

Fig 1

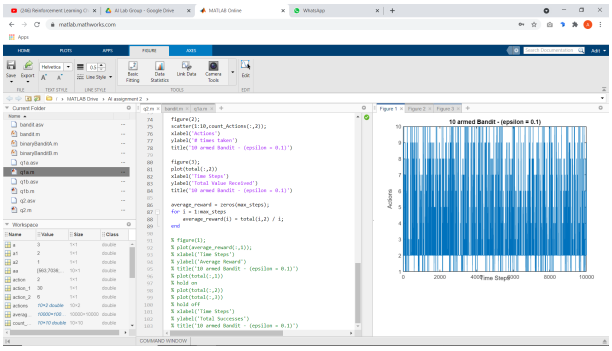
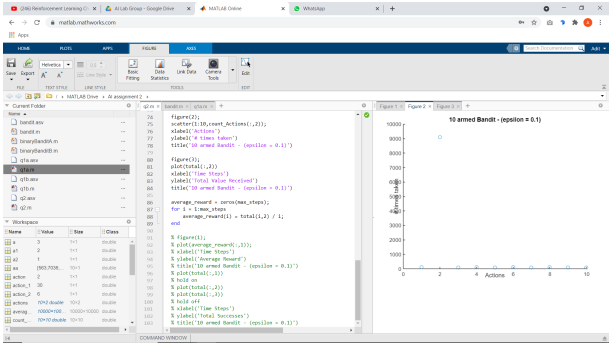


Fig 2

4.2. Solution for Part B and C

Develop a 10-armed bandit in which all ten mean-rewards start out equal and then take independent random walks (by adding a normally distributed increment with mean zero and standard deviation 0.01 to all mean-rewards on each time step). function [value] = bandit_nonstat(action) (3) The 10-armed bandit that you developed (bandit_nonstat) is difficult to crack with standard epsilon-greedy algorithm since the rewards are non-stationary. We did discuss about how to track non-stationary rewards in class. Write modified epsilon-greedy agent and show whether it is able to latch onto correct actions or not. (Try at least 10000 time steps before commenting on results)



The 10-armed bandit that you developed (bandit_nonstat) is difficult to crack with standard epsilon-greedy algorithm since the rewards are non-stationary. We did discuss about how to track non-stationary rewards in class. Write modified epsilon-greedy agent and show whether it is able to latch onto correct actions or not. (Try at least 10000 time steps before commenting on results)

Fig 3

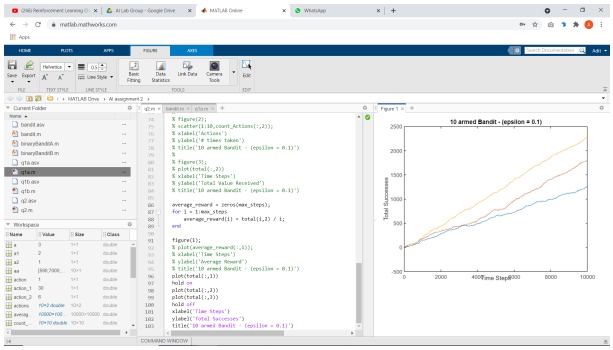


Fig 4

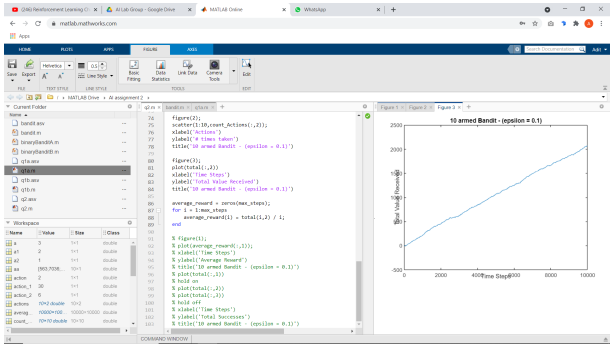
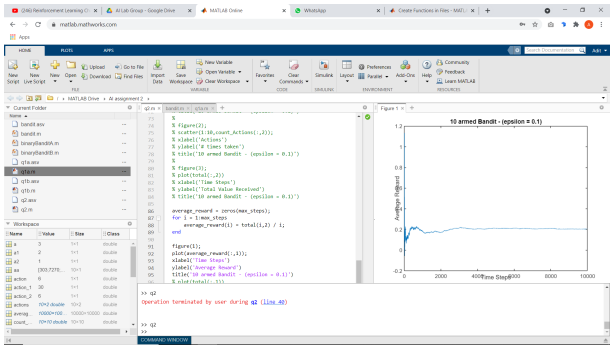


Fig 5



Hence in this assignment we tackled the binary bandits and maximized their out puts as well as constructed a 10 armed bandit with non-stationary rewards and analyzed it.

5. Assignment 10

5.1. Solution for Part A

We need to find the value function corresponding to the optimal policy using value iteration. We take the discount factor, Γ 0.99 for our implementation. An optimal policy is a policy that yields the highest expected utility. The utility of a state is the immediate reward for that state plus the expected discounted utility of the next state, assuming that the agent chooses the optimal action. That is, the utility of a state is given by:

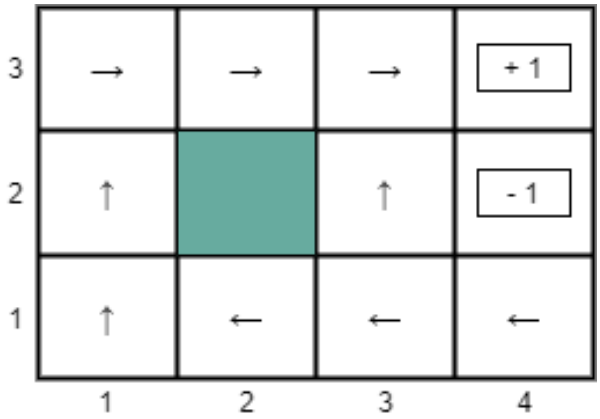
$$U(s) = R(s) + \gamma \max_{a \in A(s)} \sum_{s'} P(s' \mid s, a) U(s')$$

This is called the Bellman equation, after Richard Bellman (1957).

Value function for r(s) = -0.04 is:

3	0.812	0.868	0.918	+ 1
2	0.761		0.660	- 1
1	0.705	0.655	0.611	0.388
	1	2	3	4

Corresponding optimal policy for r(s) = -0.04 is:



Value functions for different values of immediate reward r(s) are as follows:

- r(s)=-2

3	-7.042	-4.230	-1.730	+ 1
2	-9.542		-3.570	- 1
1	-10.815	-8.474	-5.974	-3.775
	1	2	3	4

- r(s)=0.1

3	9.990	9.990	9.990	+ 1
2	9.990		9.990	- 1
1	9.990	9.990	9.990	9.990
	1	2	3	4

- r(s)=0.02

3	1.990	1.990	1.990	+ 1
2	1.990		1.990	- 1
1	1.990	1.990	1.990	1.990
	1	2	3	4

- r(s)=1

3	99.99	99.99	99.99	+ 1
2	99.99		99.99	- 1
1	99.99	99.99	99.99	99.99
	1	2	3	4

5.2. Solution for Part B

We need to answer two questions:

i. How many bicycles should we move, overnight, between each location, to maximize our total expected reward, i.e., what should be our strategy (policy) given a situation (state)?

ii. If I know this strategy, how can I compare which situations are better than others (value)?

- **Setup:** We can get two types of rewards, the first being the INR10 reward when we rent a bicycle, and the second being the -(INR2) reward for each

bicycle that we move from one location to the other. We define the Poisson distribution with class poisoon that calculates pmf values.

- **Policy Iteration Algorithm**
  A policy is a mapping from states to actions, i.e., given a state, how many bicycles should I move overnight. Now, suppose I have some policy π, then given this π, the value of a state (say s) is the expected reward that I would get when I start from s and follow π after that.
  The policy iteration algorithm consists of three components. These are:
  - **Initialization**
    We initialize the value and policy matrices arbitrarily. Given a policy, we define a value for each state, and since our state is a pair of two numbers where each number takes a value between 0 and 20, hence we represent a value by a matrix of shape (21 x 21). The policy takes a state and outputs an action.
  - **Policy Evaluation**
    Given a policy π, the value of a state (say s) is the expected reward that we would get when we start from s and follow π after that. This Bellman equation forms the basis of the value update (below):

$$V(s) \leftarrow \sum_{s',r} p(s', r|s, \pi(s))[r + \gamma V(s')]$$

Value Update

    After many such updates, V(s) converges to a number that almost satisfies (with at most some Θ error) the Bellman equation and hence represents the value of state s.
  - **Policy Improvement**
    Given a state (say s), we assign π(s) to be equal to that action that maximizes the expected reward. And we say that the policy becomes stable when none of the action maximization steps in any state causes a change in the policy. We run policy evaluation and improvement components in a loop until the policy gets stable.

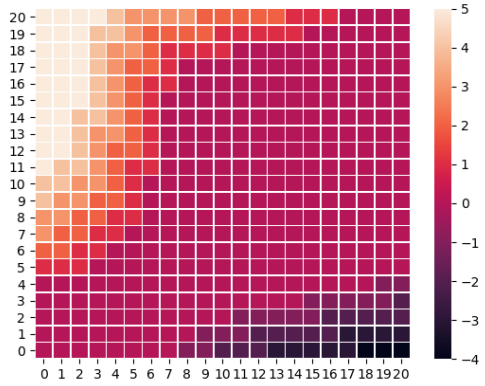
We have represented the optimal policy in form of heatmaps in order to get a graphical representation.

Resulting Policy:

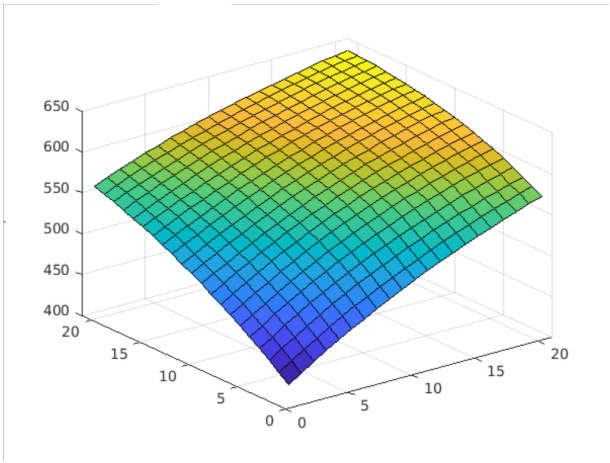
[[0 0 0 0 0 0 0 0 0 -1 -1 -2 -2 -2 -3 -3 -3 -3 -3 -4 -4 -4]
[0 0 0 0 0 0 0 0 0 0 -1 -1 -1 -2 -2 -2 -2 -2 -3 -3 -3 -3]
[0 0 0 0 0 0 0 0 0 0 0 -1 -1 -1 -1 -1 -2 -2 -2 -2 -2]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 -1 -1 -1 -1 -1 -2]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 -1 -1]
[1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[2 2 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[3 2 2 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[3 3 2 2 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[4 3 3 2 2 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[4 4 3 3 2 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[5 4 4 3 2 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[5 5 4 3 2 2 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[5 5 4 3 3 2 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[5 5 4 4 3 2 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[5 5 5 4 3 2 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[5 5 5 4 3 2 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0]
[5 5 5 4 3 2 2 1 0 0 0 0 0 0 0 0 0 0 0 0 0]
[5 5 5 4 3 3 2 1 1 1 1 0 0 0 0 0 0 0 0 0 0]
[5 5 5 4 4 3 2 2 2 2 1 1 1 1 1 0 0 0 0 0 0]
[5 5 5 5 4 3 3 3 3 2 2 2 2 2 1 1 1 0 0 0 0]]

Comparison:

Our output:



Code output:



Inference: Our formulation produces a similar output as that of the given code and produces an optimal policy.

5.3. Solution for Part B

In this case, some non-linearities are added to the original problem. These conditions can be easily added to modify the original code. We will have one more reward of -(INR4) for the second parking lot if needed.

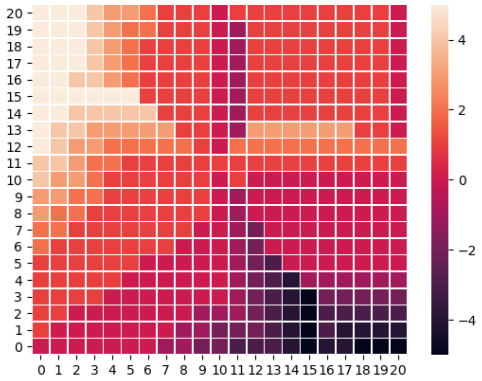
This can be accounted for with the following changes in expected_reward() function:

```
# adding reward for moving cars from one
location to another (which is negative)
if action <= 0:
    r = r + gbike.moving_reward * abs(action)
else:
    r = r + gbike.moving_reward * (action -
    1)  #one bicycle is moved by one of the
    employees for free

# adding reward for second parking lot (which
is also negative)
if new_state[0] > 10:
    r = r + gbike.second_parking_lot_reward

if new_state[1] > 10:
    r = r + gbike.second_parking_lot_reward
```

Resulting Policy:

[[0 0 0 0 0 0 0 -1 -1 -2 -2 -3 -3 -3 -4 -5 -4 -4 -5 -5 -5]
[1 0 0 0 0 0 0 0 -1 -1 -2 -2 -2 -3 -4 -5 -3 -4 -4 -4 -4]
[1 1 0 0 0 0 0 0 0 -1 -1 -1 -2 -3 -4 -5 -3 -3 -3 -3 -3]
[1 1 1 1 0 0 0 0 0 0 0 -1 -2 -3 -4 -5 -2 -2 -2 -2 -2]
[1 1 1 1 1 0 0 0 0 0 0 -1 -2 -3 -4 -1 -1 -1 -1 -1 -1]
[1 1 1 1 1 1 0 0 0 0 0 -1 -2 -3 0 0 0 0 0 0 0]
[2 1 1 1 1 1 1 1 0 0 0 -1 -2 0 0 0 0 0 0 0 0]
[2 2 1 1 1 1 1 1 1 0 0 -1 -2 0 0 0 0 0 0 0 0]
[3 2 2 1 1 1 1 1 1 1 0 -1 0 0 0 0 0 0 0 0 0]
[3 3 2 2 1 1 1 1 1 1 0 -1 0 0 0 0 0 0 0 0 0]
[4 3 3 2 1 1 1 1 1 1 0 1 0 0 0 0 0 0 0 0 0]
[4 4 3 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]
[5 4 3 3 2 2 2 2 2 1 0 2 2 2 2 2 2 2 2 2 2]
[5 4 4 3 3 3 3 3 1 1 0 -1 3 3 3 3 3 3 1 1 0]
[5 5 4 4 4 4 4 1 1 1 0 -1 1 1 1 1 1 1 1 1 0]
[5 5 5 5 5 5 1 1 1 1 0 -1 1 1 1 1 1 1 1 1 0]
[5 5 4 4 3 2 1 1 1 1 0 -1 1 1 1 1 1 1 1 1 0]
[5 5 5 4 3 2 1 1 1 1 0 -1 1 1 1 1 1 1 1 1 0]
[5 5 5 4 3 2 1 1 1 1 0 -1 1 1 1 1 1 1 1 1 0]
[5 5 5 4 3 2 2 1 1 1 0 -1 1 1 1 1 1 1 1 1 0]
[5 5 5 4 3 3 2 1 1 1 0 1 1 1 1 1 1 1 1 1 0]]

Graphical Representation:



6. Conclusion

Hence we see that Artificial Intelligence is not limited to daily life problems and is capable of outsmarting humans through its advancements, is more than capable of aiding humans in predicting, classification, and training tasks, and should be researched even further to open more possibilities and techniques.

Acknowledgment

We have taken efforts in this project, however it would not have been possible if not for Mr. Pratik Shah sir, we sincerely express our gratitude and thank you for the moral and educational support.

We would also like to acknowledge different people who have assisted in our project at various times. The internet also provided massive help for the completion of the assignment.

Last but not the least, We would like to thank our Entire team for providing great collaboration and coordination for the completion of the assignment.

References

[1] A search algorithm

[2] A* Search Algorithm