ASSIGNMENT-3
Uncertainty, Bayesian Nets, HMM and Kalman Filtering

By: Subham Maurya, 2022510

Total Buffers Used (Including All Assignments): 2/3 Days

Q1)

(a).

Direct Sampling: It generates samples directly from joint dataset distribution using the given or calculated conditional probabilities.

- ☐ **Strengths**: Simple and computationally efficient when the distribution is known. It doesn't depends on prior information or additional rules.
- ☐ **Weaknesses**: It may not work well for events with limited samples in the dataset. I.e., requires an accurate dataset that represents the distribution well.
- ☐ **Example**: Direct sampling would quickly estimate probabilities like train travelers for leisure and business because their data is available in large quantity.

Rejection Sampling: It generates samples from known distribution and accepts or rejects them based on a condition related to target distribution.

- ☐ **Strengths**: It is beneficial when the target distribution is complex or unknown.
- ☐ **Weaknesses**: Sometimes it can be computationally expensive, especially when the acceptance rate is low.
- ☐ **Example**: For estimating probabilities like P(low stress and prefers bus), rejection sampling may help in rejecting a high proportion of samples.

Gibbs Sampling: It generates samples iteratively by updating one variable at a time using their conditional distributions.

- ☐ **Strengths**: Most used and beneficial when the dataset is high-dimensional and contains complex distributions.
- ☐ **Weaknesses**: Requires conditional probabilities for each variable, which may not be easy to determine. Hence, computationally very expensive
- ☐ **Example**: Beneficial if probabilities such as P(leisure | bus), P(stress | air) etc are interdependent.

(b).

As, P(leisure | train) = 0.4

Total Number of people = 100

Amount of people who prefers train travel = 30

So, traveler for leisure = P(leisure | train) * travelers who prefers train
= 0.4 * 30
= 12

Therefore, a total of 12 travelers travel for leisure.

(c).

Let denote variable A for the person who prefers air travel and B for the person who prefers business travel.

So, P(A) = 0.8
P(B | A) = 0.2

Now,
P(A n B) = P(A) * P(B | A)
= 0.8 * 0.2
= 0.160

Therefore, the probability that a randomly selected person prefers air travel and travels for business is 0.160.

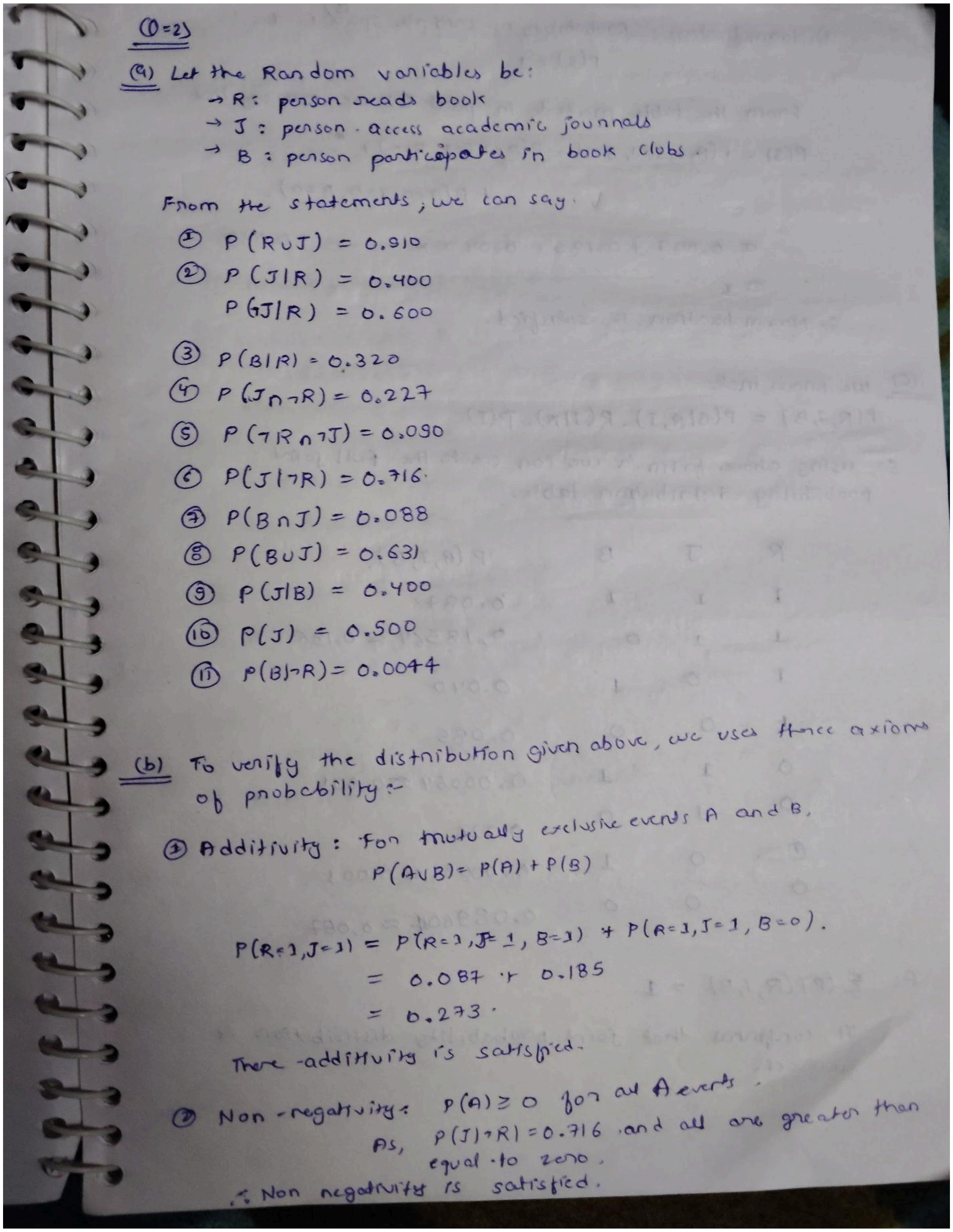
(d).

1. Accuracy: Increasing the sample size reduces sampling error and improves the reliability of estimates.

2. Precision: Precision increases as the variability of the estimate decreases.

3. Implications for the Dataset: A small dataset with only a few samples for rare events may lead to biased or imprecise estimates. Increasing sample size improves estimates for both frequent and infrequent events.

Q2)



Q-2)

(a) Let the Random variables be:

- → R : person reads book
- → J : person access academic journals
- → B : person participates in book clubs.

From the statements; we can say.

1. $P(R \cup J) = 0.910$
2. $P(J|R) = 0.400$
   $P(\neg J|R) = 0.600$
3. $P(B|R) = 0.320$
4. $P(J \cap \neg R) = 0.227$
5. $P(\neg R \cap \neg J) = 0.090$
6. $P(J|\neg R) = 0.716$.
7. $P(B \cap J) = 0.088$
8. $P(B \cup J) = 0.631$
9. $P(J|B) = 0.400$
10. $P(J) = 0.500$
11. $P(B|\neg R) = 0.0044$

(b) To verify the distribution given above, we uses three axioms of probability :-

1. Additivity : For mutually exclusive events A and B,
   $P(A \cup B) = P(A) + P(B)$

$P(R=1, J=1) = P(R=1, J=1, B=1) + P(R=1, J=1, B=0).$
$= 0.087 + 0.185$
$= 0.273.$

There additivity is satisfied.

2. Non-negativity : $P(A) \geq 0$ for all A events.
   As, $P(J|\neg R) = 0.716$ and all are greater than equal to zero.

∴ Non negativity is satisfied.

③ Normalization : Probability of sample space (S) is 1

$P(S) = 1$.

From the table maded in part (c) we can say that

$P(S) = P(R=1, J=1, B=1) + P(R=1, J=1, B=0) + - - - -$

$+ P(R=0, J=0, B=0)$.

$= 0.087 + 0.185 + 0.040 + - - - - + 0.089$

≈ 1

∴ Normalization is satisfied.

(c) We know that.

$P(R, J, B) = P(B|R, J) \cdot P(J|R) \cdot P(J)$.

So, using above formula we can create the full joint probability distribution table.

R	J	B	P (R, J, B).
1	1	1	0.087
1	1	0	$0.18564 \approx 0.186$.
1	0	1	0.040
1	0	0	0.085
0	1	1	$0.00064 \approx 0.001$
0	1	0	0.226
0	0	1	$0.0003 \approx 0.001$
0	0	0	$0.089604 \approx 0.087$

As. $\sum P(R, J, B) = 1$.

It confirms that joint probability distribution is correct.

(d) For conditional independence b/w A and B ⇒ $P(A \cap B) = P(A).P(B)$.

① Between R and J :

$P(R \cap J) = 0.273$

$P(R).P(J) = 0.398 \times 0.500$

$= 0.199$.

As. $P(R \cap J) \; != P(R).R(J)$

∴ R and J are not independent.

② Between R and B :

$P(R \cap B) = 0.1276$

$P(R).P(B) = 0.398 \times 0.128$

$= 0.0512$

As $P(R \cap B) \; != P(A).P(B)$

So, R and B are not independent.

③ Between B and J :

$P(B \cap J) = 0.088$

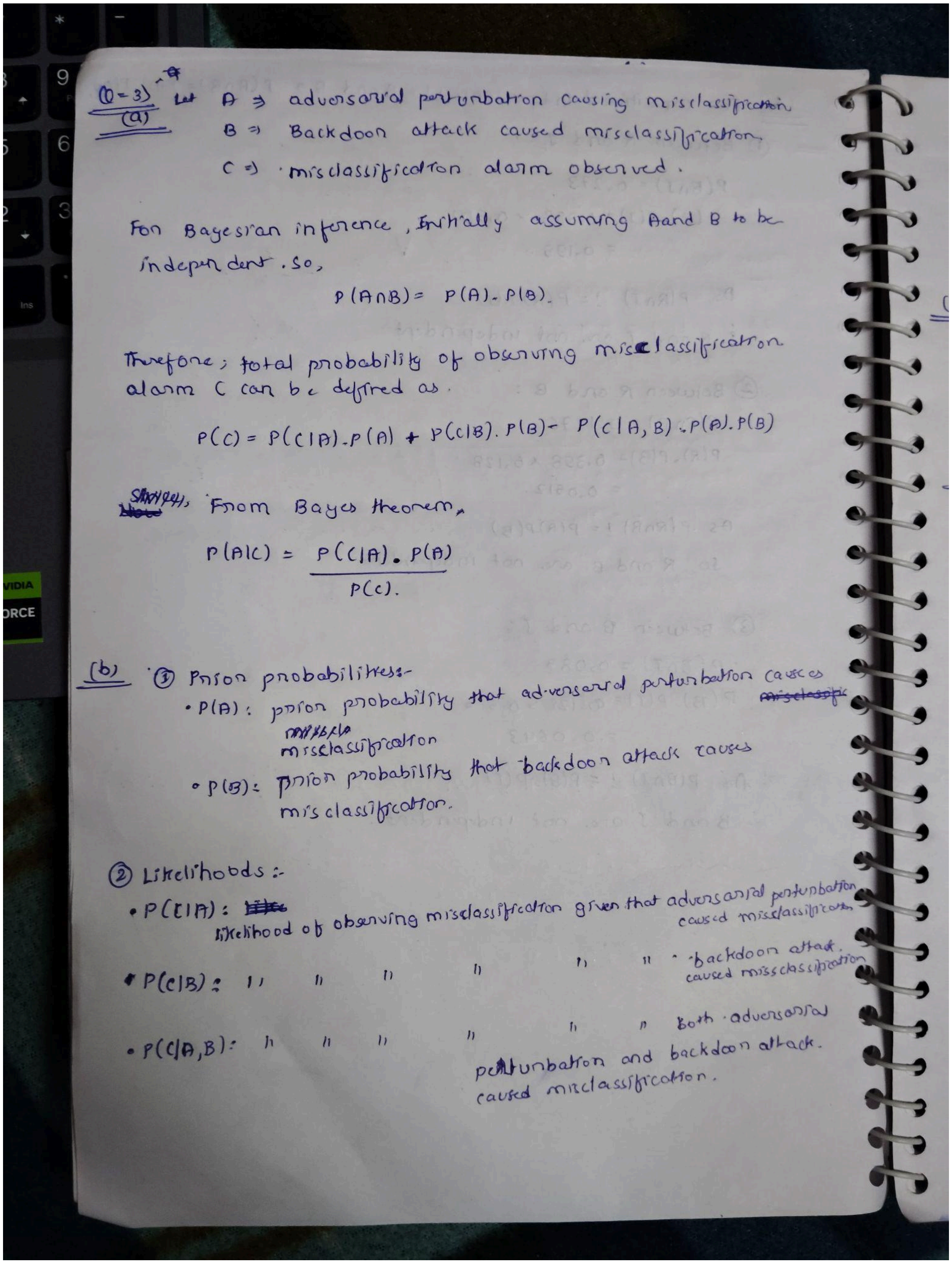
$P(B).P(J) = 0.128 \times 0.5$

$= 0.0643$

As $P(B \cap J) \; != P(B).P(J)$

∴ B and J are not independent.

Q3)



(Q-3) (a) Let A ⇒ adversarial perturbation causing misclassification

B ⇒ Backdoor attack caused misclassification.

C ⇒ misclassification alarm observed.

For Bayesian inference, Initially assuming A and B to be independent. So,

$$P(A \cap B) = P(A).P(B).$$

Therefore; total probability of observing misclassification alarm C can be defined as.

$$P(C) = P(C|A).P(A) + P(C|B).P(B) - P(C|A,B).P(A).P(B)$$

~~Now~~ Similarly, From Bayes theorem,

$$P(A|C) = \frac{P(C|A).P(A)}{P(C)}.$$

(b) ① Prior probabilities:-

- P(A): prior probability that adversarial perturbation causes ~~misclassific~~ ~~[illegible]~~ misclassification
- P(B): prior probability that backdoor attack causes misclassification.

② Likelihoods:-

- P(C|A): ~~[illegible]~~ likelihood of observing misclassification given that adversarial perturbation caused misclassification
- P(C|B): " " " " " " backdoor attack caused misclassification
- P(C|A,B): " " " " " " Both adversarial perturbation and backdoor attack caused misclassification.

③ Posterior probabilities;

- $P(A|C)$: posterior prob. that adversarial perturbation caused the misclassification given the misclassification alarm

- $P(B|C)$: " " " backdoor attack " " " " " " "

(C)

Now, the ~~new~~ new information about increasing prevalence of backdoor attacks affects our prior belief about B, which makes backdoor attack a more likely cause of misclassification alarm

Reason:

If two independent causes, A and B ~~can~~ both explain the event C, then observing alarm shifts the belief ~~[illegible]~~ towards the more likely cause.

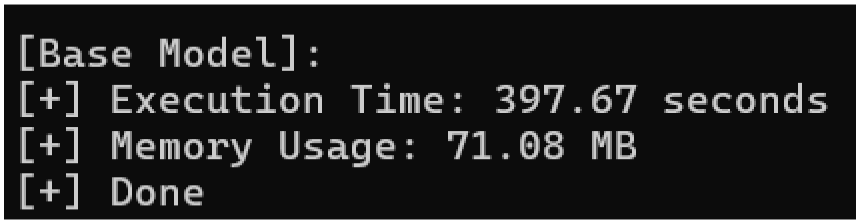
So, when $P(B)$ increases, the posterior probability $P(A|C)$ decreases. because misclassification alarm is mostly caused due to backdoor attack i.e.

$P(A|C,$ ~~[illegible]~~ and $P(B)$ increases $) < P(A|C)$.

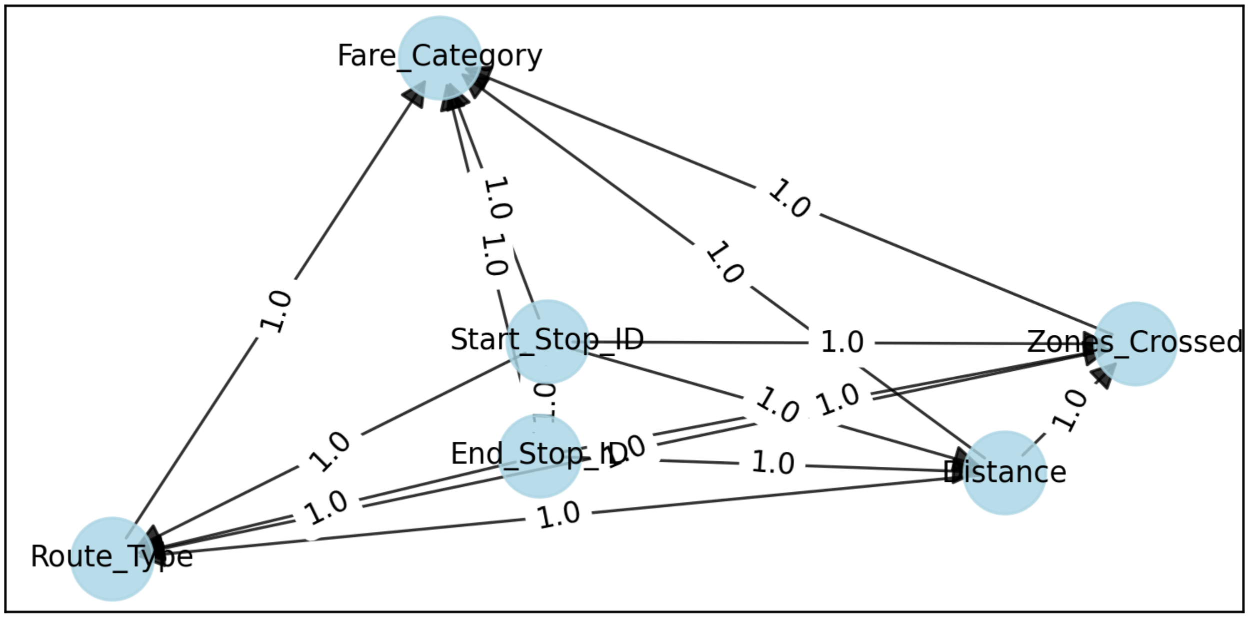
Therefore, conditioning on increases prevalence of backdoor triggers makes it less likely that adversarial perturbations caused misclassification.

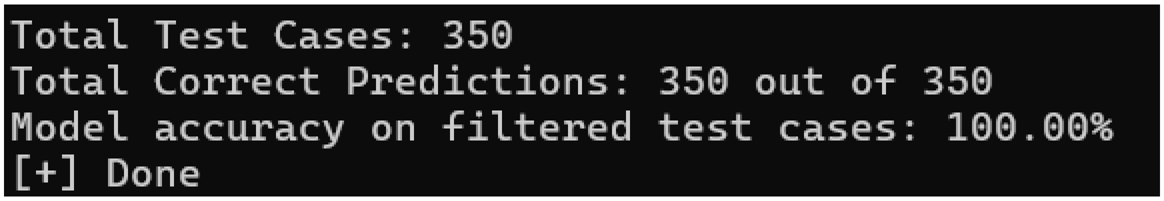
Q4)

(a) Base Model:



```
[Base Model]:
[+] Execution Time: 397.67 seconds
[+] Memory Usage: 71.08 MB
[+] Done
```

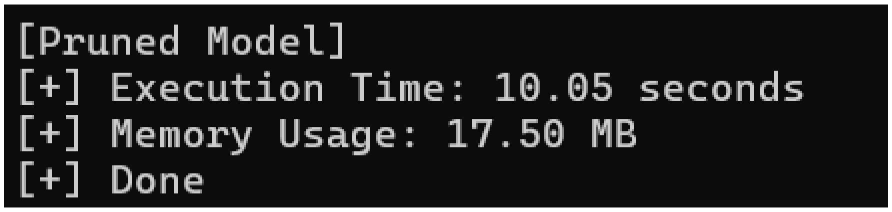




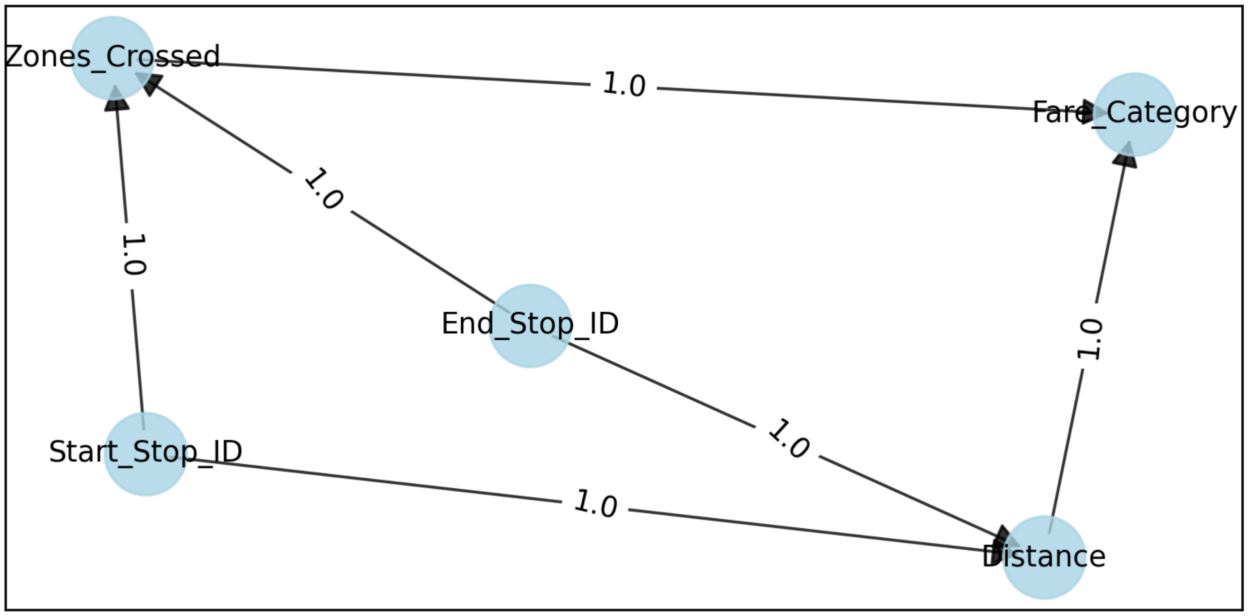
```
Total Test Cases: 350
Total Correct Predictions: 350 out of 350
Model accuracy on filtered test cases: 100.00%
[+] Done
```

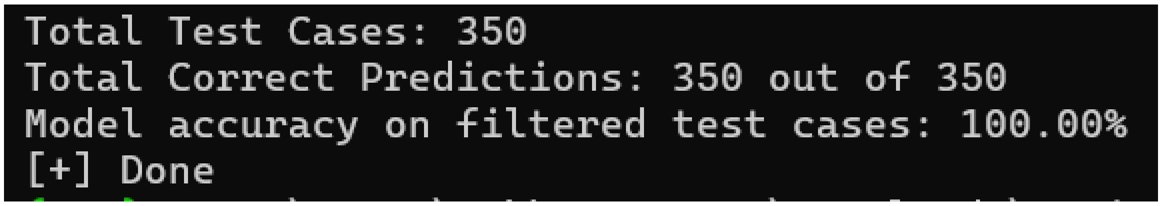
The base model is the basic DAG which is created by observing the given dataset. It contains a total of n*(n-1) edges where n is the total number of nodes/features available. It can be seen that it takes a lot of execution time and memory because it is the beginner-level DAG that is created and can be further optimized which is discussed in the next part. Apart from that, the accuracy of this model is 100%.

(b) Pruned Model:



```
[Pruned Model]
[+] Execution Time: 10.05 seconds
[+] Memory Usage: 17.50 MB
[+] Done
```

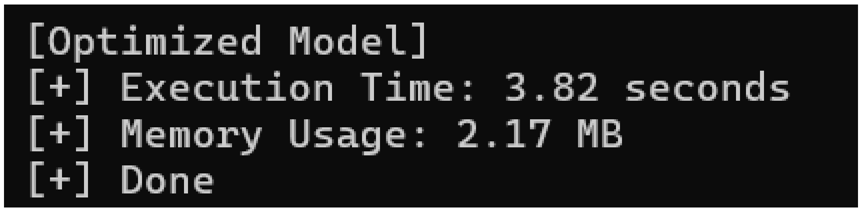




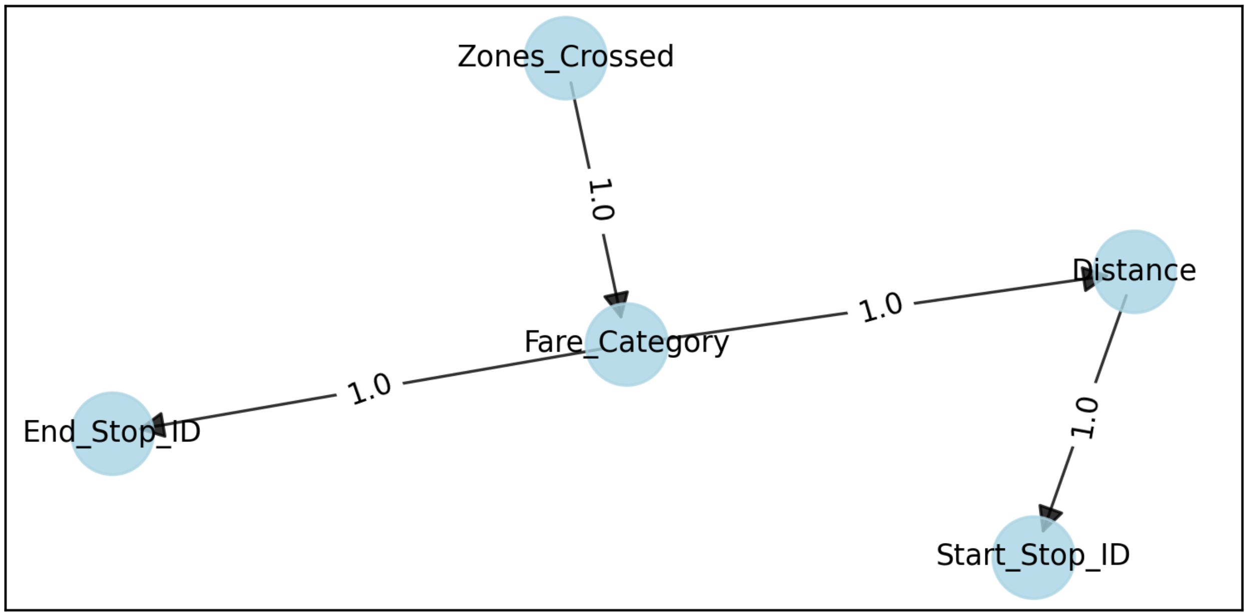
```
Total Test Cases: 350
Total Correct Predictions: 350 out of 350
Model accuracy on filtered test cases: 100.00%
[+] Done
```

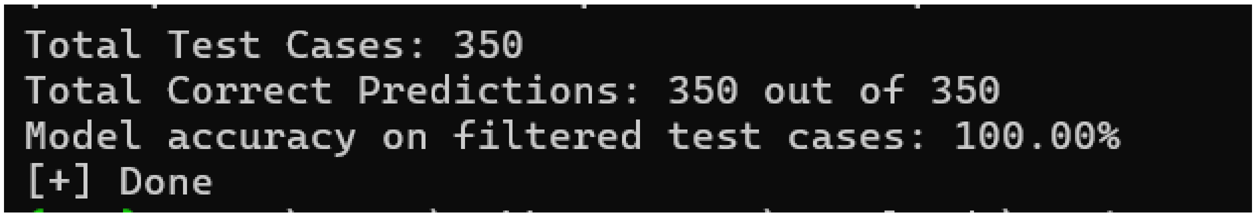
From the given features, we used the edges pruning and node pruning techniques to prune some of the edges and nodes(Route_type) from the graph because it greatly reduces the size of the original Bayesian network. This type of pruning technique reduces the time required to fit the model by maintaining the same precision and efficiency which can be seen from the observation of execution time and memory usage in the above parts.

(c) Optimized Model



```
[Optimized Model]
[+] Execution Time: 3.82 seconds
[+] Memory Usage: 2.17 MB
[+] Done
```



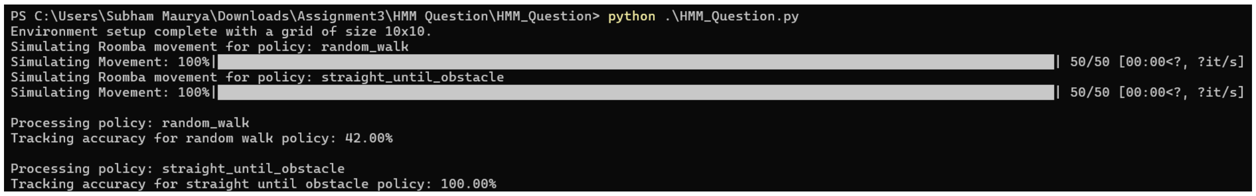


```
Total Test Cases: 350
Total Correct Predictions: 350 out of 350
Model accuracy on filtered test cases: 100.00%
[+] Done
```

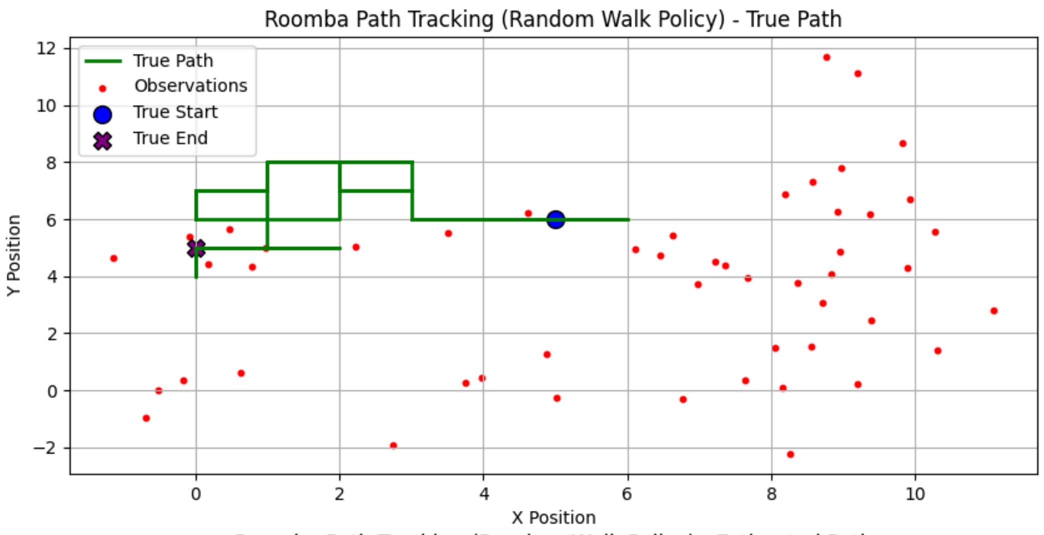
In comparison to the original Bayesian network (A), the optimized Bayesian network(C) contains few nodes and edges which eventually reduces the time required to fit the model along with minimizing the memory usage which can be seen from the above observations. Hence, it increases the efficiency of the model by providing 100% accuracy by utilizing less resources.

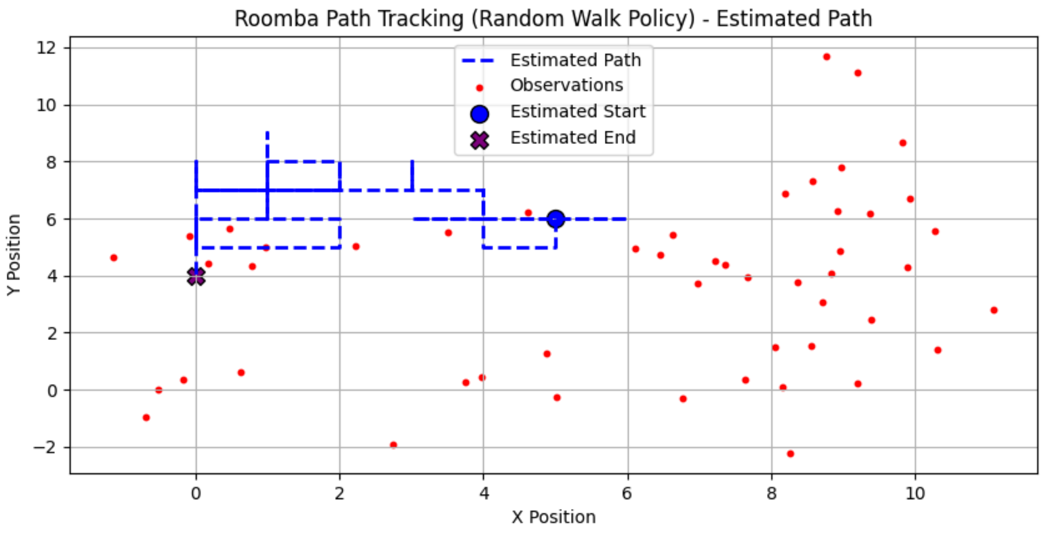
Q5)

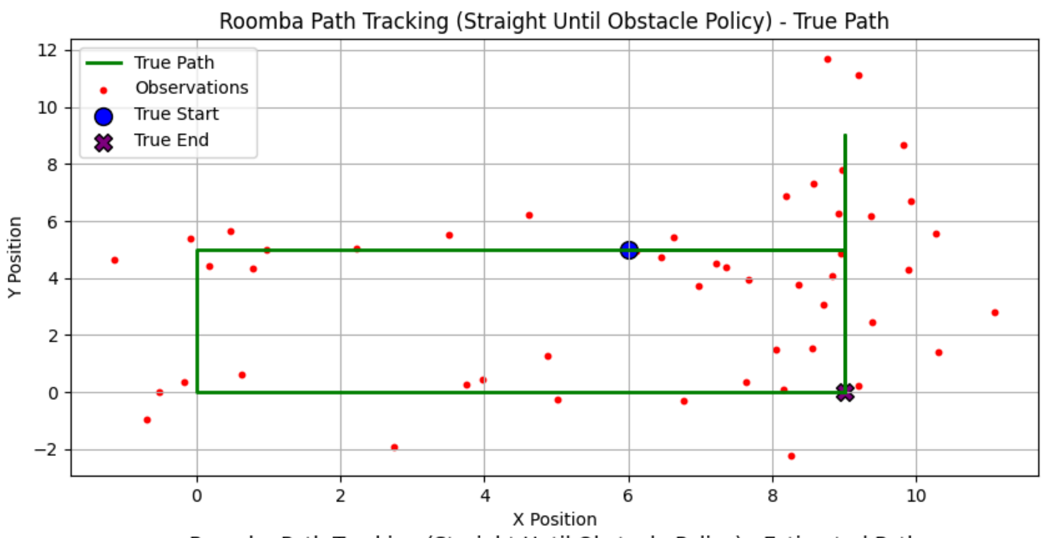
(a) Seed = 111

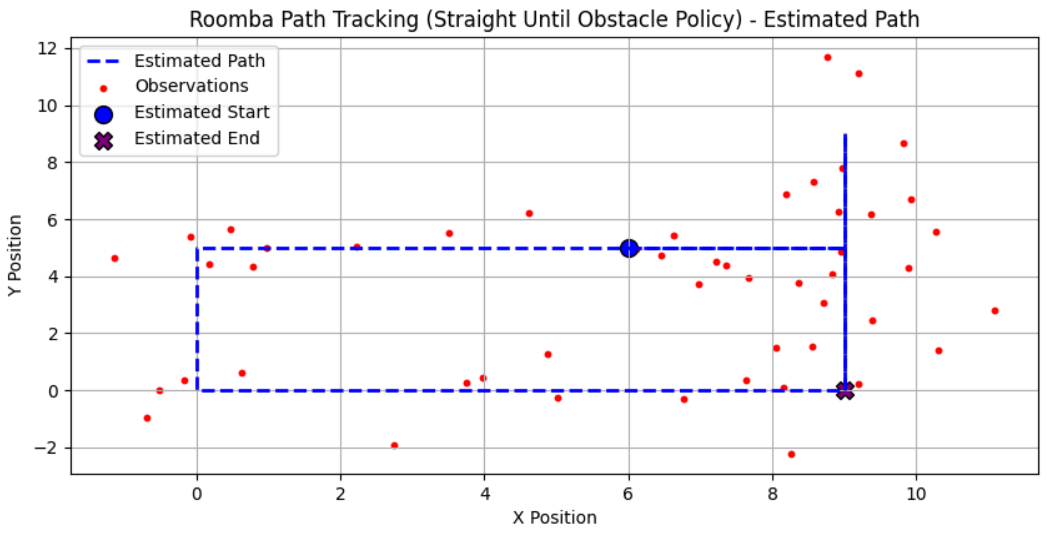


```
PS C:\Users\Subham Maurya\Downloads\Assignment3\HMM Question\HMM_Question> python .\HMM_Question.py
Environment setup complete with a grid of size 10x10.
Simulating Roomba movement for policy: random_walk
Simulating Movement: 100%|████████████████████████████████████████████████████████| 50/50 [00:00<?, ?it/s]
Simulating Roomba movement for policy: straight_until_obstacle
Simulating Movement: 100%|████████████████████████████████████████████████████████| 50/50 [00:00<?, ?it/s]

Processing policy: random_walk
Tracking accuracy for random walk policy: 42.00%

Processing policy: straight_until_obstacle
Tracking accuracy for straight until obstacle policy: 100.00%
```



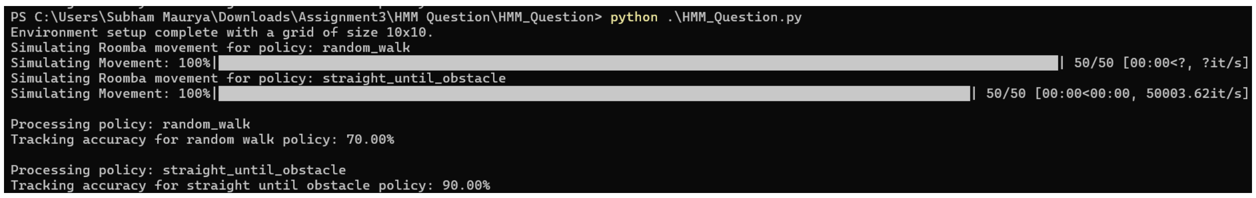




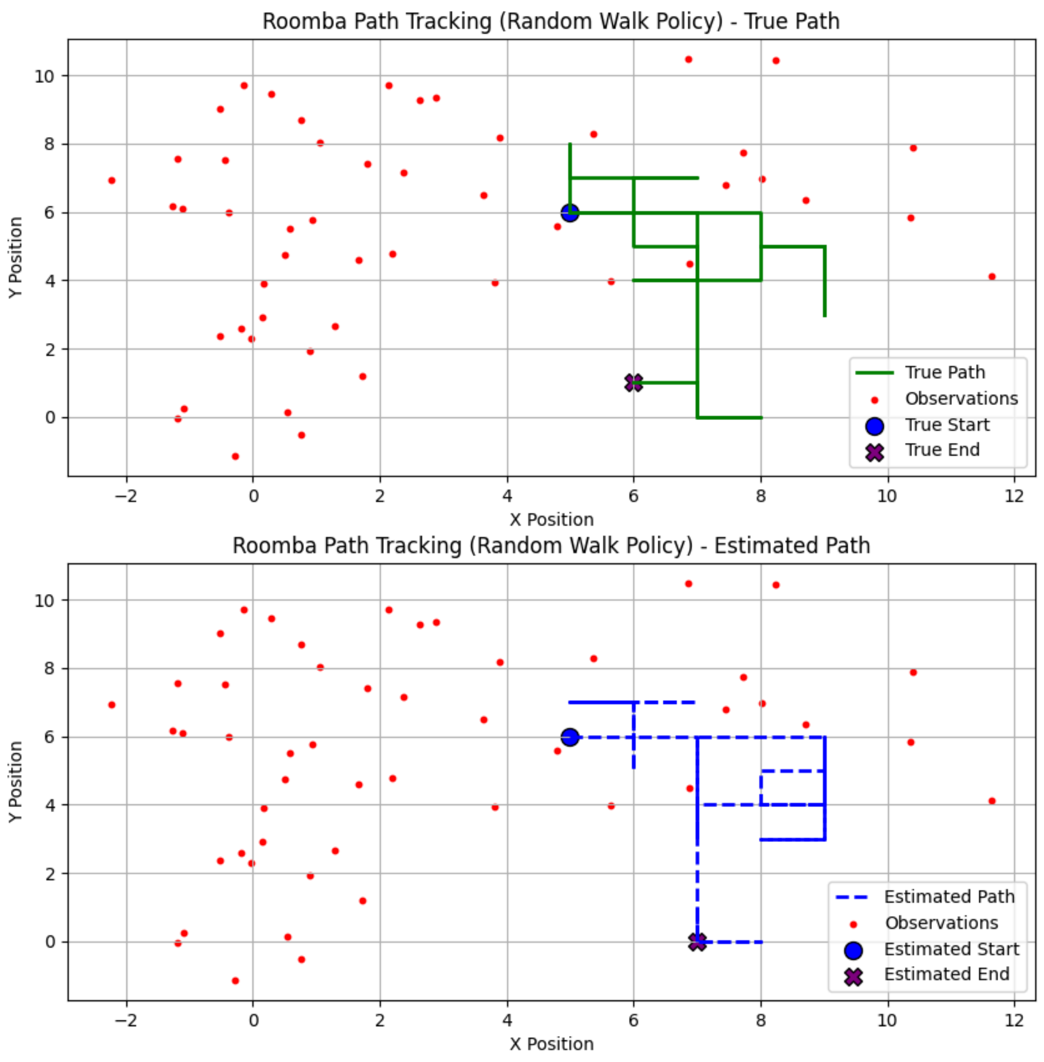


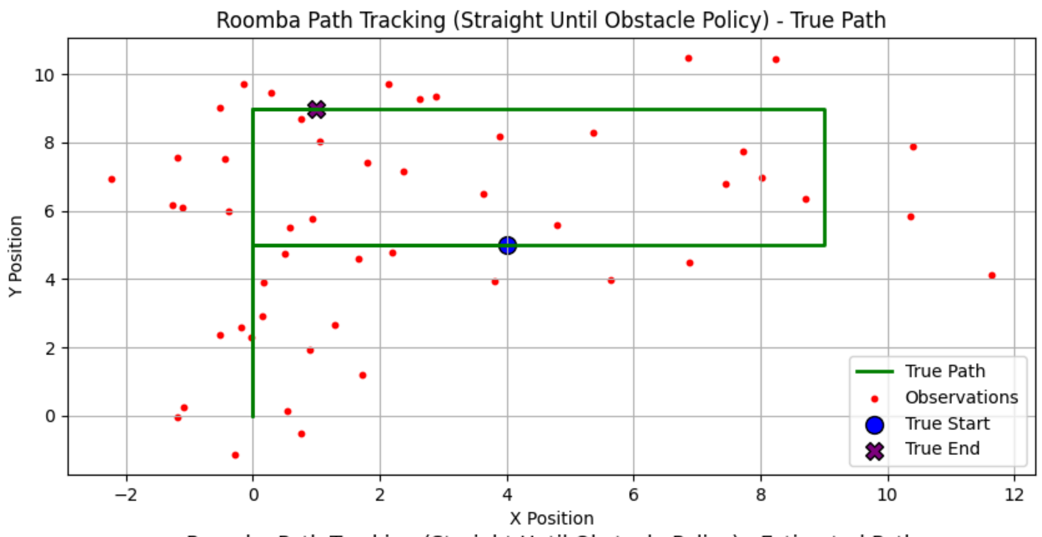
For seed = 111, the best accuracy is 100% which is given by straight until obstacle policy

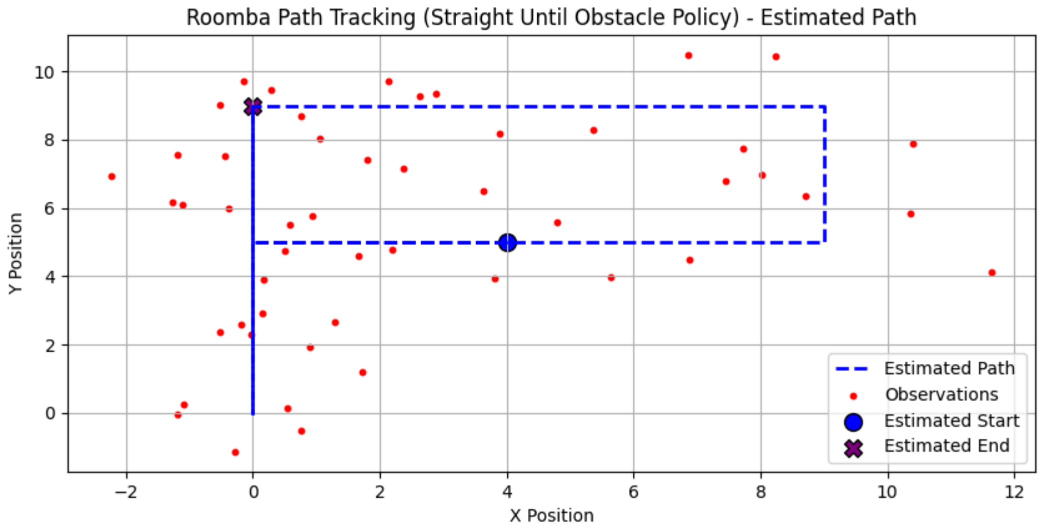
(b) Seed = 60



```
PS C:\Users\Subham Maurya\Downloads\Assignment3\HMM Question\HMM_Question> python .\HMM_Question.py
Environment setup complete with a grid of size 10x10.
Simulating Roomba movement for policy: random_walk
Simulating Movement: 100%|████████████████████████████████████████| 50/50 [00:00<?, ?it/s]
Simulating Roomba movement for policy: straight_until_obstacle
Simulating Movement: 100%|████████████████████████████████████| 50/50 [00:00<00:00, 50003.62it/s]

Processing policy: random_walk
Tracking accuracy for random walk policy: 70.00%

Processing policy: straight_until_obstacle
Tracking accuracy for straight until obstacle policy: 90.00%
```

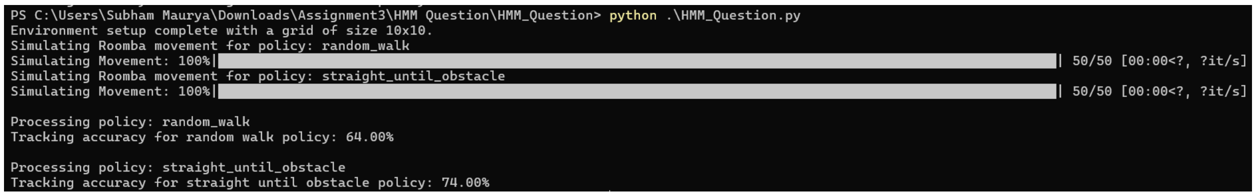




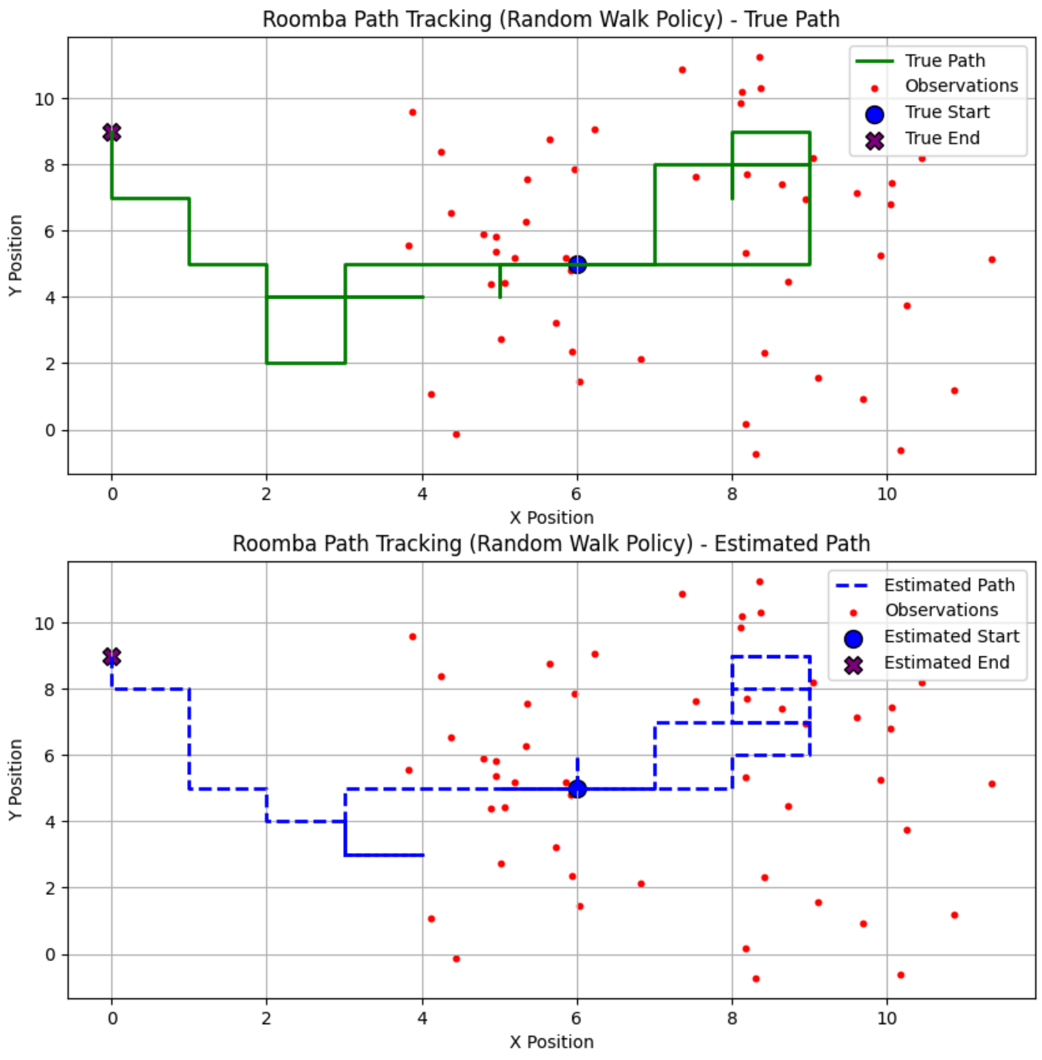


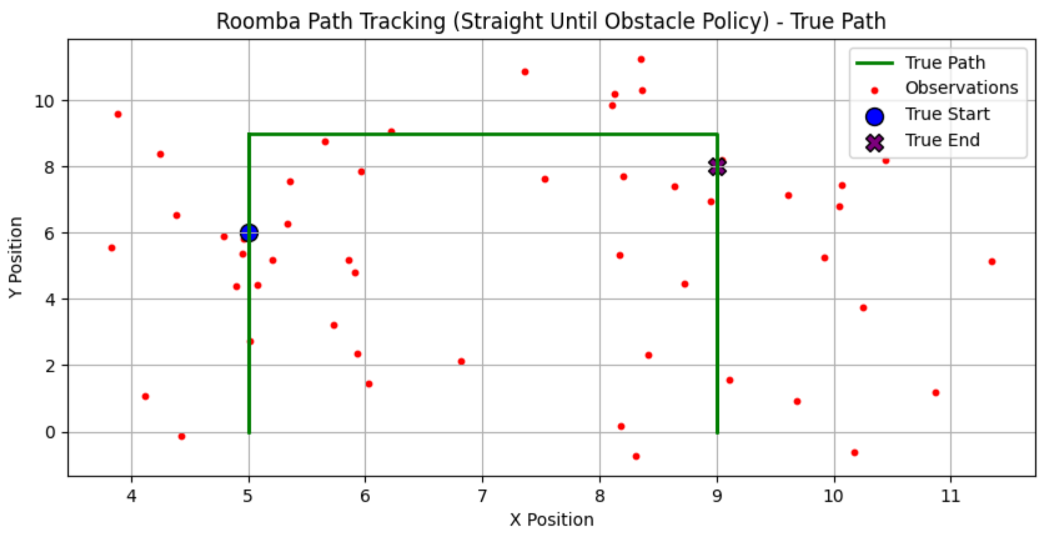
For seed = 60, the best accuracy is 90% which is given by straight until obstacle policy

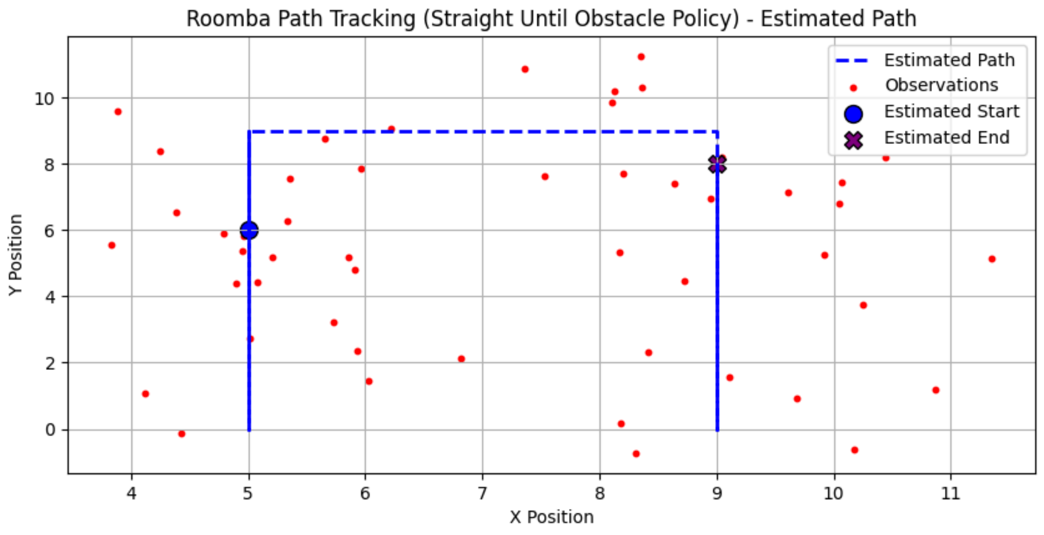
(c) Seed = 120



```
PS C:\Users\Subham Maurya\Downloads\Assignment3\HMM Question\HMM_Question> python .\HMM_Question.py
Environment setup complete with a grid of size 10x10.
Simulating Roomba movement for policy: random_walk
Simulating Movement: 100%|████████████████████████████████████████| 50/50 [00:00<?, ?it/s]
Simulating Roomba movement for policy: straight_until_obstacle
Simulating Movement: 100%|████████████████████████████████████████| 50/50 [00:00<?, ?it/s]

Processing policy: random_walk
Tracking accuracy for random walk policy: 64.00%

Processing policy: straight_until_obstacle
Tracking accuracy for straight until obstacle policy: 74.00%
```







For seed = 120, the best accuracy is 74% which is again given by straight until obstacle policy.

From the above observations, we can say that the **Straight Until Obstacle** policy results in better accuracy in estimating Roomba's path because of the following reasons:

1. Predictability of Movement: In the **Straight Until Obstacle** policy, the Roomba moves in a straight line until it encounters an obstacle, at which it changes direction. This movement behavior has fewer state transitions per time step, reducing the complexity of the transition model. Consequently, the likelihood of estimating the correct sequence of states is higher since Roomba's next state is often straightforward to predict.

2. Fewer Sudden Changes in Heading: Unlike the **Random Walk** policy, which involves frequent changes in direction, the **Straight Until Obstacle** policy maintains consistent headings/directions for longer durations.

3. Noise Filtering with Structured Movement: Noisy observations are more easily aligned with the true path when movement is structured (straight lines). Hence, If the observation noise deviates slightly from the true position, the straight movement pattern helps "pull" the estimated path back to alignment with the true path.

4. Improved Transition Model The **Straight Until Obstacle** policy creates a more deterministic transition model given a state and heading, the next state is almost deterministic unless an obstacle is encountered. In contrast, the **Random Walk** policy involves multiple equally likely transitions, increasing the uncertainty in the state sequence.