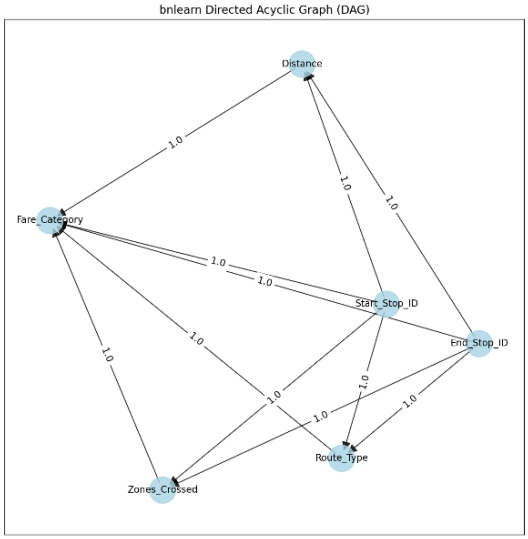
AI Assignment-03 Report

Coding Assignment

Problem: 01

Task: 1
Edge Relation is found best on the basis of logical relations between each and every pair possible, also maintaining the basic property of the graph that it should follow, which is a Directed Acyclic Graph.



Edges:

- Start_Stop_ID -> Distance
- End_Stop_ID -> Distance
- Start_Stop_ID -> Zones_Crossed
- End_Stop_ID -> Zones_Crossed
- Start_Stop_ID -> Route_Type
- End_Stop_ID -> Route_Type
- Start_Stop_ID -> Fare_Category
- End_Stop_ID -> Fare_Category
- Distance->Fare -> Category
- Zones_Crossed -> Fare_Category
- Route_Type -> Fare_Category

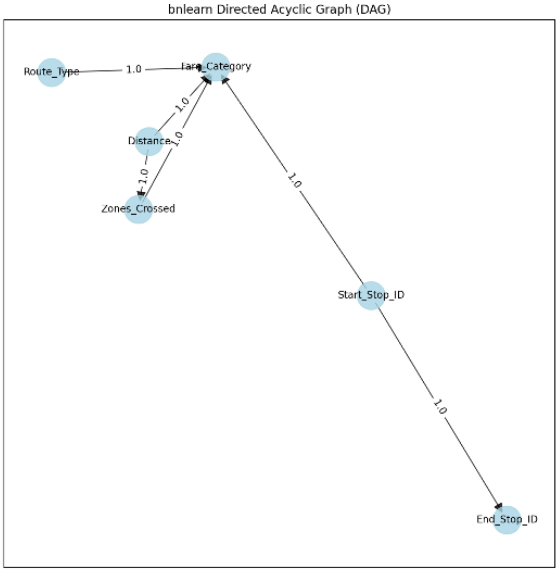
```
Model saved base_model.pkl
Time of Execution to fit:  868.8139162063599
```

Got 100% accuracy

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

Task: 2
Edges:

- Start_Stop_ID -> End_Stop_ID
- Start_Stop_ID -> Fare_Category
- Distance -> Fare_Category
- Zones_Crossed -> Fare_Category
- Route_Type -> Fare_Category
- Distance -> Zones_Crossed



Edge Pruning Reasoning:
Applying Random pruning to simplify the Bayesian Network by removing a subset of edges randomly was irrespective of the contribution of edges to the target variable, whether it was direct or indirect. This reduced the complexity of the model and brought faster training times and runtime efficiency. Although random pruning did not take into consideration the significance of the edges, the number of conditional dependencies was notably decreased in contrast, and the CPTs became simpler with low memory usage. This trade-off offered a fair accuracy of prediction while improving computational performance.

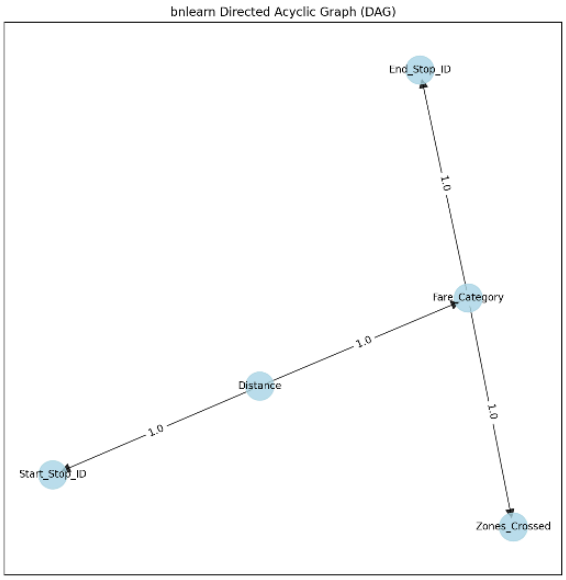
Took much less time the base model, where it took 868s it only took 22s. Got same results 100% accuracy

```
Model saved pruned_model.pkl
Time of Execution to pruned:  22.021241426467896
```

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

Task: 3
Edges:

Fare_Category -> End_Stop_ID
Distance -> Fare_Category
Distance -> Start_Stop_ID
Fare_Category -> Zones_Crossed



Took even less time to execute. Got accuracy of 100%

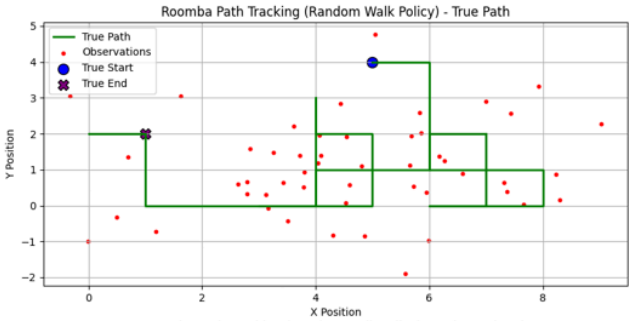
```
Model saved optimized_model.pkl
Time of Execution to optimized:  13.634575366973877
```

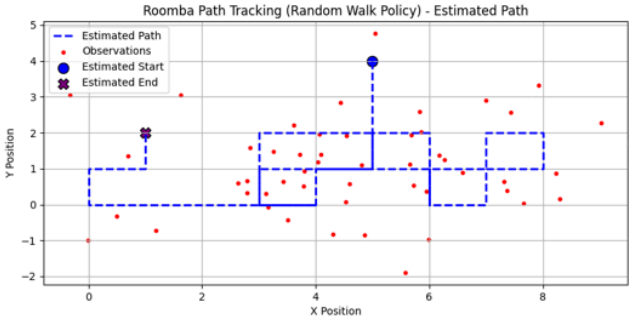
```
Total Test Cases: 350
Total Correct Predictions: 350 out of 350
Model accuracy on filtered test cases: 100.00%
```

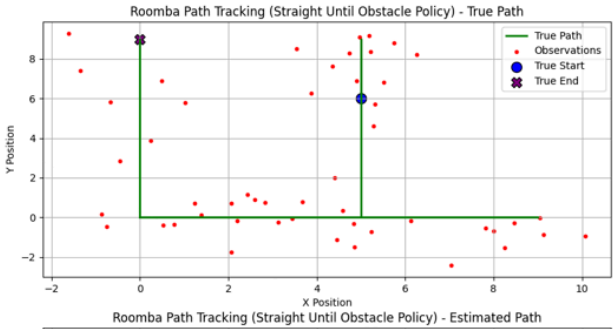
Problem 03

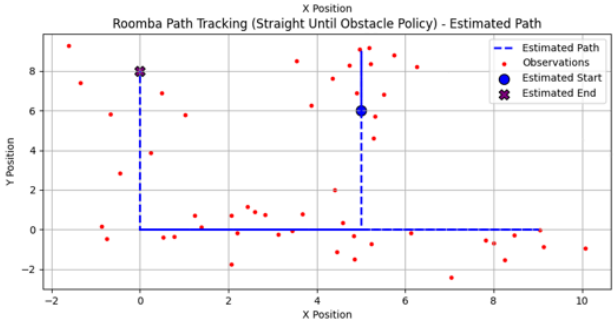
The best result in terms of accuracy I found for the Straight Until Obstacle policy, On average, this policy has given a better accuracy from the Random Walk Policy
.
Seed Values: [235, 42, 100, 30], For all screenshots of results attached from the next page

```
For seed 235
Environment setup complete with a grid of size 10x10.
Simulating Roomba movement for policy: random_walk
Simulating Roomba movement for policy: straight_until_obstacle

Processing policy: random_walk
Simulating Movement: 100%|██████████| 50/50 [00:00<?, ?it/s]
Simulating Movement: 100%|██████████| 50/50 [00:00<?, ?it/s]
Tracking accuracy for random walk policy: 24.00%

Processing policy: straight_until_obstacle
Tracking accuracy for straight until obstacle policy: 46.00%
```

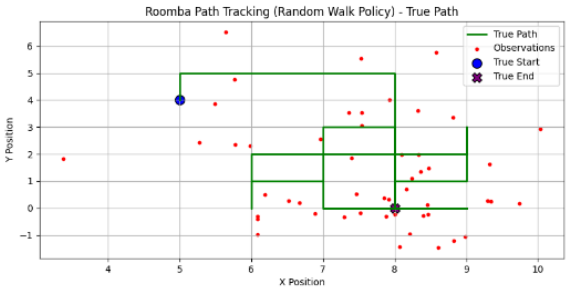


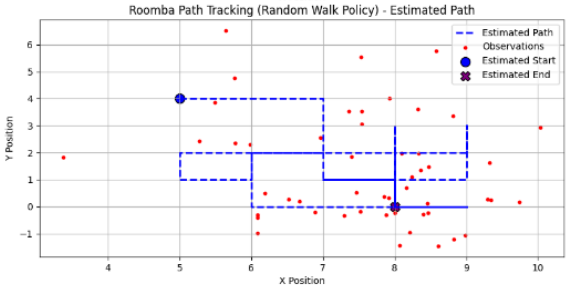


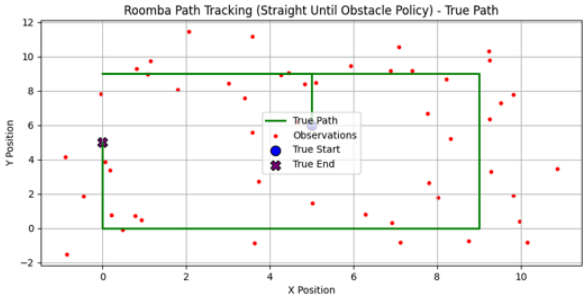


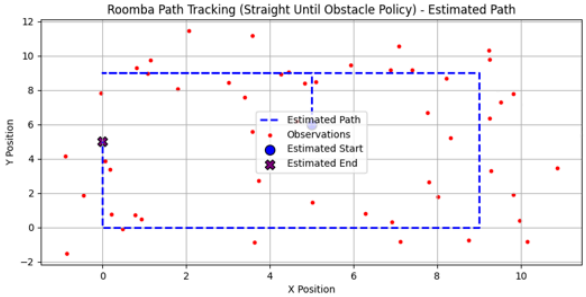



```
For seed 42
Environment setup complete with a grid of size 10x10.
Simulating Roomba movement for policy: random_walk
Simulating Movement: 100%|██████████| 50/50 [00:00<00:00, 49683.77it/s]
Simulating Movement: 100%|██████████| 50/50 [00:00<?, ?it/s]
Simulating Roomba movement for policy: straight_until_obstacle

Processing policy: random_walk
Tracking accuracy for random walk policy: 58.00%

Processing policy: straight_until_obstacle
Tracking accuracy for straight until obstacle policy: 72.00%
```

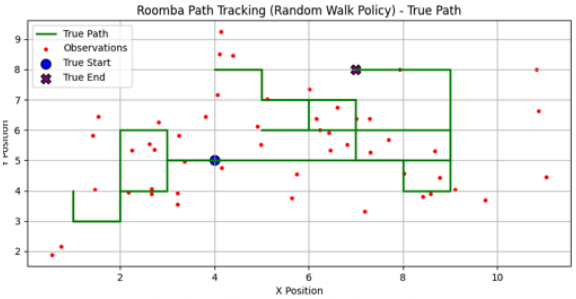


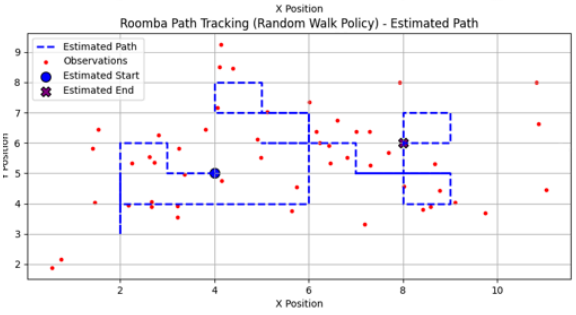


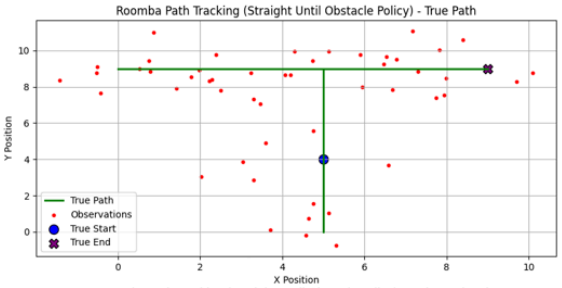


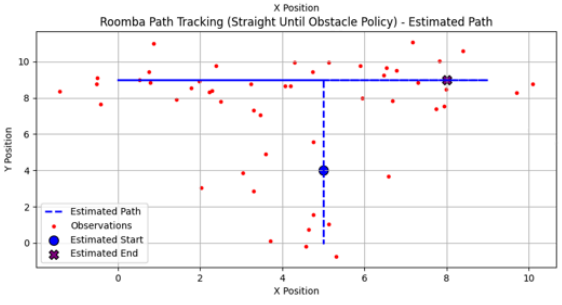



```
For seed 100
Environment setup complete with a grid of size 10x10.
Simulating Roomba movement for policy: random_walk
Simulating Roomba movement for policy: straight_until_obstacle

Processing policy: random_walk
Simulating Movement: 100%|██████████| 50/50 [00:00<?, ?it/s]
Simulating Movement: 100%|██████████| 50/50 [00:00<?, ?it/s]
Tracking accuracy for random walk policy: 32.00%

Processing policy: straight_until_obstacle
Tracking accuracy for straight until obstacle policy: 34.00%
```

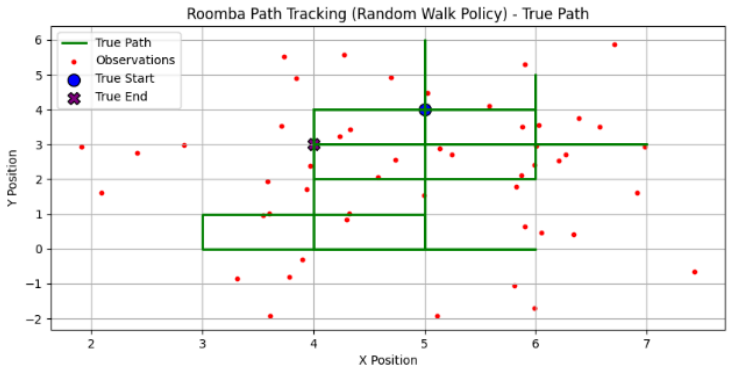


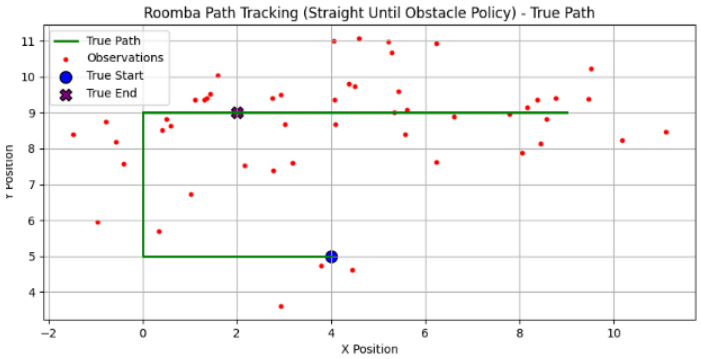


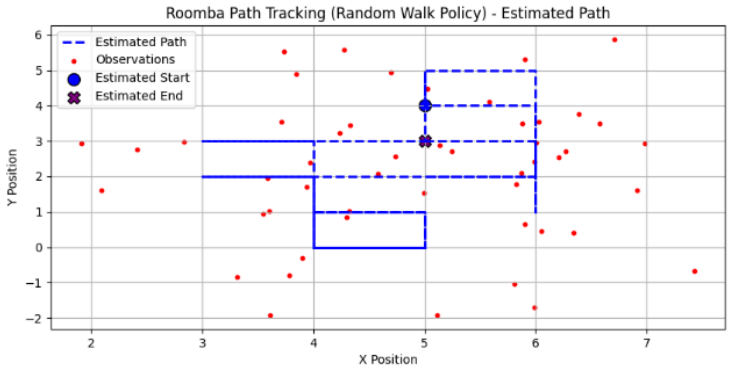


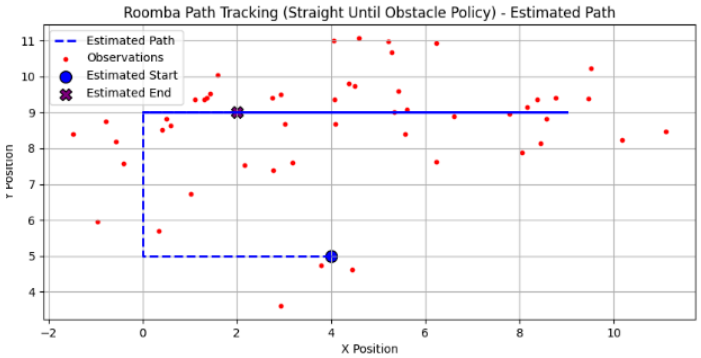


```
For seed 30
Environment setup complete with a grid of size 10x10.
Simulating Roomba movement for policy: random_walk
Simulating Movement: 100%|██████████| 50/50 [00:00<00:00, 48377.21it/s]
Simulating Movement: 100%|██████████| 50/50 [00:00<?, ?it/s]
Simulating Roomba movement for policy: straight_until_obstacle

Processing policy: random_walk
Tracking accuracy for random walk policy: 38.00%

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









Estimates Paths.csv

	Seed	Policy	Estimated Path
1	235	random_walk	[((5, 4), 'N'), ((5, 3), 'N'), ((…
2	235	straight_until_obstacle	[((5, 6), 'S'), ((5, 7), 'S'), ((…
3	100	random_walk	[((4, 5), 'W'), ((4, 5), 'N'), ((…
4	100	straight_until_obstacle	[((5, 4), 'N'), ((5, 3), 'N'), ((…
5	42	random_walk	[((5, 4), 'N'), ((5, 4), 'E'), ((…
6	42	straight_until_obstacle	[((5, 6), 'S'), ((5, 7), 'S'), ((…
7	30	random_walk	[((5, 4), 'N'), ((5, 4), 'S'), ((…
8	30	straight_until_obstacle	[((4, 5), 'W'), ((3, 5), 'W'), ((…