Ex. No.4 **BAYESIAN NETWORKS**

Date:

Aim:

To write a python program to implement a Bayesian network for the Monty Hall problem.

Algorithm:

Step 1. Start by importing the required libraries such as math and pomegranate.

Step 2. Define the discrete probability distribution for the guest's initial choice of door

Step 3. Define the discrete probability distribution for the prize door

Step 4. Define the conditional probability table for the door that Monty picks based on the guest's choice and the prize door

Step 5. Create State objects for the guest, prize, and Monty's choice

Step 6. Create a Bayesian Network object and add the states and edges between them

Step 7. Bake the network to prepare for inference

Step 8. Use the predict_proba method to calculate the beliefs for a given set of evidence

Step 9. Display the beliefs for each state as a string.

Step 10. Stop

Program:

```
import math
from pomegranate import *

# Initially the door selected by the guest is completely random
guest = DiscreteDistribution({'A': 1./3, 'B': 1./3, 'C': 1./3})

# The door containing the prize is also a random process
prize = DiscreteDistribution({'A': 1./3, 'B': 1./3, 'C': 1./3})

# The door Monty picks, depends on the choice of the guest and the prize door
monty = ConditionalProbabilityTable(
    [['A', 'A', 'A', 0.0],
     ['A', 'A', 'B', 0.5],
     ['A', 'A', 'C', 0.5],
     ['A', 'B', 'A', 0.0],
    ['A', 'B', 'B', 0.0],
    ['A', 'B', 'C', 1.0],
    ['A', 'C', 'A', 0.0],
    ['A', 'C', 'B', 1.0],
    ['A', 'C', 'C', 0.0],
    ['B', 'A', 'A', 0.0],
    ['B', 'A', 'B', 0.0],
    ['B', 'A', 'C', 1.0],
    ['B', 'B', 'A', 0.5],
    ['B', 'B', 'B', 0.0],
    ['B', 'B', 'C', 0.5],
    ['B', 'C', 'A', 1.0],
    ['B', 'C', 'B', 0.0],
    ['B', 'C', 'C', 0.0],
    ['C', 'A', 'A', 0.0],
    ['C', 'A', 'B', 1.0],
    ['C', 'A', 'C', 0.0],
    ['C', 'B', 'A', 1.0],
    ['C', 'B', 'B', 0.0],
    ['C', 'B', 'C', 0.0],
    ['C', 'C', 'A', 0.5],
    ['C', 'C', 'B', 0.5],
    ['C', 'C', 'C', 0.0]], [guest, prize])

d1 = State(guest, name="guest")
d2 = State(prize, name="prize")
d3 = State(monty, name="monty")

# Building the Bayesian Network
network = BayesianNetwork("Solving the Monty Hall Problem With Bayesian Networks")
network.add_states(d1, d2, d3)
network.add_edge(d1, d3)
network.add_edge(d2, d3)
network.bake()

# Compute the probabilities for each scenario
beliefs = network.predict_proba({'guest': 'A'})
print("\n".join("{}\t{}".format(state.name, str(belief)) for state, belief in zip(network.states,
beliefs)))

beliefs = network.predict_proba({'guest': 'A', 'monty': 'B'})
print("\n".join("{}\t{}".format(state.name, str(belief)) for state, belief in zip(network.states,
beliefs)))

beliefs = network.predict_proba({'guest': 'A', 'prize': 'B'})
print("\n".join("{}\t{}".format(state.name, str(belief)) for state, belief in zip(network.states,
beliefs)))
```

Viva Questions:

1. What is a Bayesian network and how does it work?
2. What are the key differences between Bayesian networks and other probabilistic models such as Naive Bayes or Markov Networks?
3. What is the purpose of the directed edges in a Bayesian network and how are they used to perform probabilistic inference?
4. Can you discuss some of the challenges in constructing Bayesian networks and how they can be addressed?
5. What are some real-world applications of Bayesian networks and how have they been used in these applications?

Result:

Thus, the Python program for implementing Bayesian Networks was successfully developed and the output was verified.