Name – Omkar Shivarkar
PRN no - 121A9057

Batch - B3
Branch - AIML

Experiment 9

Aim:- Write a program for Bayesian Networks.

Program:-

```python
import pgmpy.models
import pgmpy.inference
import networkx as nx
import pylab as plt
import matplotlib.pylab as pylab

# Create a Bayesian network
model = pgmpy.models.BayesianNetwork([('Burglary', 'Alarm'),
                                      ('Earthquake', 'Alarm'),
                                      ('Alarm', 'JohnCalls'),
                                      ('Alarm', 'MaryCalls')])

# Define conditional probability distributions (CPD)
cpd_burglary = pgmpy.factors.discrete.TabularCPD('Burglary', 2, [[0.001], [0.999]])
cpd_earthquake = pgmpy.factors.discrete.TabularCPD('Earthquake', 2, [[0.002], [0.998]])
cpd_alarm = pgmpy.factors.discrete.TabularCPD('Alarm', 2, [[0.95, 0.94, 0.29, 0.001],
                                                           [0.05, 0.06, 0.71, 0.999]],
                                                           evidence=['Burglary', 'Earthquake'],
                                                           evidence_card=[2, 2])

cpd_john = pgmpy.factors.discrete.TabularCPD('JohnCalls', 2, [[0.90, 0.05],
                                                              [0.10, 0.95]],
                                                              evidence=['Alarm'],
                                                              evidence_card=[2])
```

```python
cpd_mary = pgmpy.factors.discrete.TabularCPD('MaryCalls', 2, [[0.70, 0.01],
                                                              [0.30, 0.99]],
                                                              evidence=['Alarm'],
                                                              evidence_card=[2])


# Add CPDs to the network structure
model.add_cpds( *cpds: cpd_burglary, cpd_earthquake, cpd_alarm, cpd_john, cpd_mary)

# Check if the model is valid
assert model.check_model()
```

```python
# Print probability distributions
print('Probability distribution, P(Burglary):')
print(cpd_burglary)
print()
print('Probability distribution, P(Earthquake):')
print(cpd_earthquake)
print()
print('Joint probability distribution, P(Alarm | Burglary, Earthquake):')
print(cpd_alarm)
print()
print('Joint probability distribution, P(JohnCalls | Alarm):')
print(cpd_john)
print()
print('Joint probability distribution, P(MaryCalls | Alarm):')
print(cpd_mary)
print()

# Plot the model
pos = nx.shell_layout(model)
nx.draw(model, pos, with_labels=True, node_size=3000, node_color='skyblue', font_size=10, font_color='black', font_weight='bold')
plt.savefig( *args: 'alarm.png', format='PNG')

plt.show()

# Perform variable elimination for inference
infer = pgmpy.inference.VariableElimination(model)
```

```python
# Calculate the probability of burglary if John and Mary call (0: True, 1: False)
posterior_probability = infer.query(variables=['Burglary'], evidence={'JohnCalls': 0, 'MaryCalls': 0})
print('Posterior probability of Burglary if JohnCalls(True) and MaryCalls(True):')
print(posterior_probability)
print()

# Calculate the probability of the alarm sounding if there is a burglary and an earthquake (0: True, 1: False)
posterior_probability = infer.query(variables=['Alarm'], evidence={'Burglary': 0, 'Earthquake': 0})
print('Posterior probability of Alarm sounding if Burglary(True) and Earthquake(True):')
print(posterior_probability)
```

Output:-

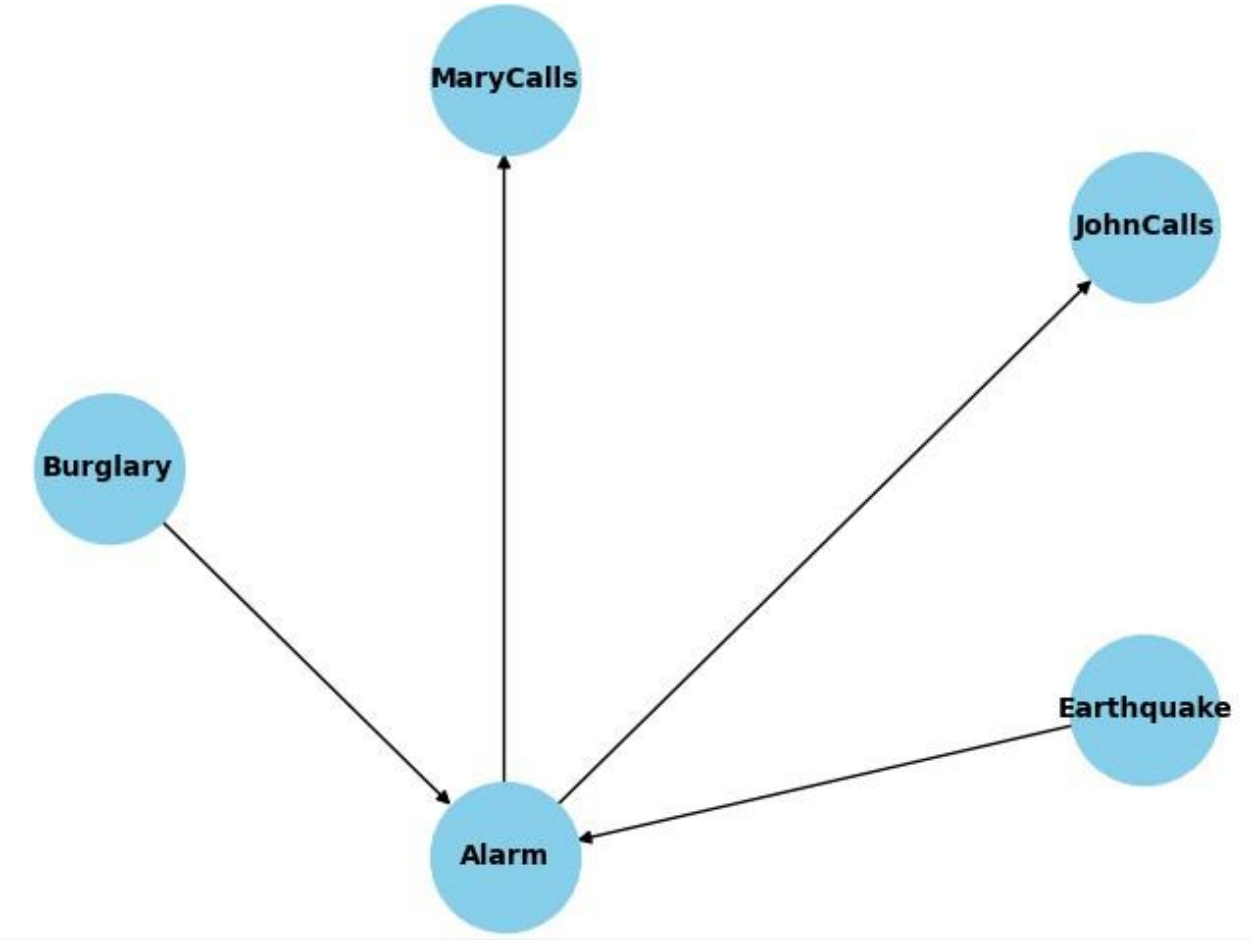
```
Probability distribution, P(Burglary):
+-------------+-------+
| Burglary(0) | 0.001 |
+-------------+-------+
| Burglary(1) | 0.999 |
+-------------+-------+
```

```
Probability distribution, P(Earthquake):
+--------------+-------+
| Earthquake(0) | 0.002 |
+--------------+-------+
| Earthquake(1) | 0.998 |
+--------------+-------+

Joint probability distribution, P(Alarm | Burglary, Earthquake):
+------------+--------------+--------------+--------------+--------------+
| Burglary   | Burglary(0)  | Burglary(0)  | Burglary(1)  | Burglary(1)  |
+------------+--------------+--------------+--------------+--------------+
| Earthquake | Earthquake(0) | Earthquake(1) | Earthquake(0) | Earthquake(1) |
+------------+--------------+--------------+--------------+--------------+
| Alarm(0)   | 0.95         | 0.94         | 0.29         | 0.001        |
+------------+--------------+--------------+--------------+--------------+
| Alarm(1)   | 0.05         | 0.06         | 0.71         | 0.999        |
+------------+--------------+--------------+--------------+--------------+

Joint probability distribution, P(JohnCalls | Alarm):
+--------------+----------+----------+
| Alarm        | Alarm(0) | Alarm(1) |
+--------------+----------+----------+
| JohnCalls(0) | 0.9      | 0.05     |
+--------------+----------+----------+
| JohnCalls(1) | 0.1      | 0.95     |
+--------------+----------+----------+

Joint probability distribution, P(MaryCalls | Alarm):
+--------------+----------+----------+
| Alarm        | Alarm(0) | Alarm(1) |
+--------------+----------+----------+
| MaryCalls(0) | 0.7      | 0.01     |
+--------------+----------+----------+
| MaryCalls(1) | 0.3      | 0.99     |
+--------------+----------+----------+
```



Conclusion:- Hence, we have successfully implemented the program for Bayesian Networks.