Name : Aryan Deepak Saraf
Div : D20B
Roll No. : 49

AI&DS2 Experiment 01

<u>Aim :</u> To Implement Inferencing with Bayesian Network in python

<u>Theory :</u>
A Bayesian Network, also known as a Belief Network, is a probabilistic graphical model used to compute uncertainties by using the concept of probability. It uses a Directed Acyclic Graph (DAG) to represent a set of variables and their conditional dependencies. In a DAG, nodes represent variables and links or edges represent the relationships between them. A key feature of a DAG is that following the directions of the edges will never lead to a closed loop.

Key Concepts

- Joint Probability: This is a statistical measure of the probability of two or more events happening at the same time. It can be represented as the probability of the intersection of these events, for example, P(A,B,C).
- Conditional Probability: This is the probability of an event X occurring given that another event Y has already occurred.
- Bayesian Network Formula: The joint probability distribution of all variables in a Bayesian Network can be formulated as the product of the conditional probabilities of each variable given its parents. The formula is:

  $$P(X_1, ..., X_n) = \Pi_{i=1}^{n} p(X_i | Parents(X_i))$$

  Here, Xi is a random variable, and Parents(Xi) refers to the parent nodes of Xi. The probability of a variable depends on its parents.

Example of a Bayesian Network: Daily Habits and Alertness
Consider a network that models the relationships between a person's daily habits and their level of alertness. The model includes four variables: Sleep, Stress, Coffee, and Alert. The variables and their states are defined as follows:

Sleep	0 = Poor	1 = Good
Stress	0 = High	1 = Low
Coffee	0 = No	1 = Yes
Alert	0 = Not Alert	1 = Alert

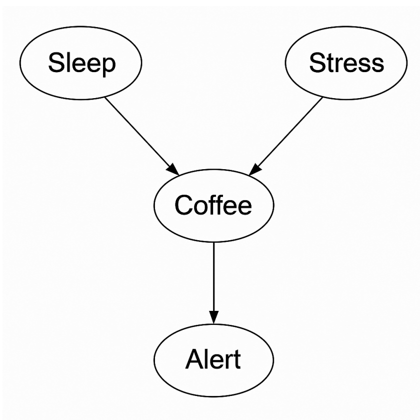
This network is structured as a Directed Acyclic Graph (DAG) that shows the dependencies between these variables. The Sleep and Stress nodes are the parent variables, as they are not dependent on any other factors in this model. The decision to have Coffee depends on both the quality of Sleep and the level of Stress. Finally, the level of Alertness is directly influenced by whether the person has had Coffee.

This structure results in the following joint probability distribution:

P(Sleep,Stress,Coffee,Alert) = P(Alert|Coffee) ∗ P(Coffee|Sleep,Stress) ∗ P(Sleep) ∗ P(Stress)

- P(Sleep) is the unconditional probability of getting a certain quality of sleep.
- P(Stress) is the unconditional probability of having a certain level of stress.
- P(Coffee|Sleep,Stress) is the conditional probability of consuming coffee given a specific quality of sleep and level of stress.
- P(Alert|Coffee) is the conditional probability of being alert given whether or not coffee was consumed.

These probabilities are defined in the code using Conditional Probability Tables (CPDs), which specify the likelihood of each variable's state given the states of its parent variables. This allows the model to calculate the probability of any variable, such as alertness, when other variables are known.



This image shows the variables and their relationships, along with conditional probability tables (CPTs) that define the probabilities for each variable given its parents.

Code :

STEP 1: Install Required Library

```
!pip install pgmpy
```

STEP 2: Import Libraries

```
from pgmpy.models import DiscreteBayesianNetwork
from pgmpy.factors.discrete import TabularCPD
from pgmpy.inference import VariableElimination
```

STEP 3: Define Bayesian Network Structure

```
model = DiscreteBayesianNetwork([
    ('Sleep', 'Coffee'),
    ('Stress', 'Coffee'),
    ('Coffee', 'Alert')
])
```

STEP 4: Define CPDs (Conditional Probability Tables)

```
# CPD for Sleep: 0 = Poor, 1 = Good
cpd_sleep = TabularCPD(variable='Sleep', variable_card=2, values=[[0.4], [0.6]])

# CPD for Stress: 0 = High, 1 = Low
cpd_stress = TabularCPD(variable='Stress', variable_card=2, values=[[0.5], [0.5]])

# CPD for Coffee: 0 = No, 1 = Yes
cpd_coffee = TabularCPD(variable='Coffee', variable_card=2,
                        values=[[0.9, 0.6, 0.5, 0.2],    # No coffee
                                [0.1, 0.4, 0.5, 0.8]],   # Yes coffee
                        evidence=['Sleep', 'Stress'],
                        evidence_card=[2, 2])

# CPD for Alert: 0 = Not Alert, 1 = Alert
cpd_alert = TabularCPD(variable='Alert', variable_card=2,
                       values=[[0.7, 0.2],    # Not alert
                               [0.3, 0.8]],   # Alert
                       evidence=['Coffee'],
                       evidence_card=[2])
```

STEP 5: Add CPDs and Check Model

```
model.add_cpds(cpd_sleep, cpd_stress, cpd_coffee, cpd_alert)

# Validate model
print("Model check:", model.check_model())
```

```
Model check: True
```

STEP 6: Perform Inference

```
inference = VariableElimination(model)

# What is the probability of being alert given good sleep and low stress?
result = inference.query(variables=['Alert'], evidence={'Sleep': 1, 'Stress': 1})
print(result)
```

```
+----------+--------------+
| Alert    |   phi(Alert) |
+==========+==============+
| Alert(0) |       0.3000 |
+----------+--------------+
| Alert(1) |       0.7000 |
+----------+--------------+
```

Conclusion : We successfully modeled a real-life scenario using Bayesian Networks in Python. By defining variables like sleep, stress, coffee intake, and alertness, we used a Directed Acyclic Graph and CPTs to compute conditional probabilities. This helped us perform inference and understand how daily habits influence alertness.

49_Prac_01