Exercise No. 4			
Topic:	**Module 2.0: Probabilistic Reasoning in AI**	**Week No.**	7-9
Course Code:	**CSST101**	**Term:**	1st Semester
Course Title:	**Advance Representation and Reasoning**	**Academic Year:**	2024-2025
Student Name		**Section**	
Due date		**Points**	

Topic 2.2: Bayesian Networks

Implementation of Bayesian Networks in Google Colab

Exercise 1: Setting Up the Environment

1. **Install the Required Library:**
   - Start by installing pgmpy in your Google Colab environment.

```
!pip install pgmpy
```

2. **Import Libraries:**
   - Import the necessary libraries.

```
import numpy as np
import pandas as pd
from pgmpy.models import BayesianModel
from pgmpy.inference import VariableElimination
from pgmpy.inference import BeliefPropagation
from pgmpy.factors.discrete import TabularCPD
```

Exercise 2: Building a Simple Bayesian Network

1. Define the Structure:

- Create a Bayesian Network for the following variables: Weather (Sunny, Rainy), Traffic (Light, Heavy), and Late (On Time, Late). Define the relationships:
  - Traffic depends on Weather.
  - Late depends on Traffic.

```
# Define the structure of the Bayesian Network
model = BayesianModel([('Weather', 'Traffic'), ('Traffic', 'Late')])
```

2. Define Conditional Probability Tables (CPTs):

- Use **TabularCPD** to define the CPDs for each variable.

```
# Weather CPD
cpd_weather = TabularCPD(variable='Weather', variable_card=2, values=[[0.8], [0.2]])

# Traffic CPD given Weather
cpd_traffic = TabularCPD(variable='Traffic', variable_card=2,
                         values=[[0.9, 0.5], [0.1, 0.5]],  # P(Light | Sunny, Rainy)
                         evidence=['Weather'],
                         evidence_card=[2])

# Late CPD given Traffic
cpd_late = TabularCPD(variable='Late', variable_card=2,
                      values=[[0.95, 0.4], [0.05, 0.6]],  # P(On Time | Light, Heavy)
                      evidence=['Traffic'],
                      evidence_card=[2])

# Add CPDs to the model
model.add_cpds(cpd_weather, cpd_traffic, cpd_late)

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

Exercise 3: Querying the Bayesian Network

1. **Perform Exact Inference:**
    - Use the variable elimination method to determine the probability of being Late given that it is Rainy.

```
# Create an inference object
inference = VariableElimination(model)

# Query the probability of being Late given that Weather is Rainy
result = inference.query(variables=['Late'], evidence={'Weather': 1})  # 1 corresponds
print(result)
```

Exercise 4: Parameter Learning

1. **Simulate a Dataset:**
    - Create a synthetic dataset of observations for Weather, Traffic, and Late.

```
# Create a synthetic dataset
# 0 for Sunny, 1 for Rainy
data = pd.DataFrame({
    'Weather': np.random.choice([0, 1], size=1000, p=[0.8, 0.2]),
    'Traffic': np.nan,
    'Late': np.nan
})

# Fill Traffic based on Weather
data.loc[data['Weather'] == 0, 'Traffic'] = np.random.choice(
    [0, 1],
    size=data[data['Weather'] == 0].shape[0],
    p=[0.9, 0.1]
)

data.loc[data['Weather'] == 1, 'Traffic'] = np.random.choice(
    [0, 1],
    size=data[data['Weather'] == 1].shape[0],
    p=[0.5, 0.5]
)

# Fill Late based on Traffic
data['Late'] = np.where(
    data['Traffic'] == 0,
    np.random.choice([0, 1], size=data.shape[0], p=[0.95, 0.05]),
    np.random.choice([0, 1], size=data.shape[0], p=[0.4, 0.6])
)
```

2. Estimate the Parameters:

- Use the dataset to estimate the CPDs for the Traffic and Late nodes.

```
from pgmpy.estimators import MaximumLikelihoodEstimator

# Create a Bayesian Model
model = BayesianModel([('Weather', 'Traffic'), ('Traffic', 'Late')])

# Fit the model to the data using Maximum Likelihood Estimation
model.fit(data, estimator=MaximumLikelihoodEstimator)

# Check the estimated CPDs
for cpd in model.get_cpds():
    print(cpd)
```

Exercise 5: Visualizing the Bayesian Network

1. **Visualize the Network Structure:**
    - Use the networkx library to visualize the Bayesian Network.

```
import matplotlib.pyplot as plt
import networkx as nx

# Convert the Bayesian Model to a NetworkX graph
nx_graph = model.to_networkx()

# Draw the graph
plt.figure(figsize=(8, 6))
pos = nx.spring_layout(nx_graph)
nx.draw(
    nx_graph, pos,
    with_labels=True, node_color='lightblue',
    font_weight='bold', arrows=True
)
plt.title('Bayesian Network Structure')
plt.show()
```

Submission Format:

- Upload your Python scripts and Notebook to your GitHub repository.
- Ensure the repository is well-organized, with folders and files clearly labeled (e.g., scripts/, notebooks/, README.md)
- Filename [SECTION]-[SURNAME]-EXER4 e.g. 3A-BERNARDINO-EXER4

Rubric for Exercises on Bayesian Networks Using Google Colab

Criteria	Excellent (4 points)	Good (3 points)	Fair (2 points)	Poor (1 point)
Exercise 1: Setting Up the Environment	Libraries are installed correctly, and all necessary libraries are imported without errors.	Minor issues in library imports; however, the main functionality is intact.	Libraries are partially installed or imported, causing some functionality issues.	Libraries are not installed or imported, leading to multiple errors.
Exercise 2: Building a Simple Bayesian Network	Bayesian Network structure is defined correctly with all relationships accurately represented. All CPTs are correctly implemented and added to the model.	The structure is mostly correct; however, one or two relationships or CPTs may be slightly incorrect.	The structure has significant errors, with major relationships or CPTs incorrectly defined.	The structure is not defined, or there are critical errors in all CPTs.
Exercise 3: Querying the Bayesian Network	Exact inference is performed correctly, with accurate query results printed. Code is clean and well-organized.	Inference is mostly correct, but minor errors exist in query formulation or result interpretation.	Inference is attempted, but significant errors lead to incorrect results or unclear code.	Inference is not performed or completely incorrect, leading to no results.
Exercise 4: Parameter Learning	A synthetic dataset is generated correctly, and parameters are estimated accurately using Maximum Likelihood Estimation. All outputs are well-documented.	The dataset is mostly generated correctly, but minor issues may affect parameter estimation. Outputs are documented but lack some clarity.	Dataset generation has significant issues, affecting parameter estimation. Outputs are poorly documented.	Dataset is not generated, and parameter estimation is absent or incorrect.
Exercise 5: Visualizing the Bayesian Network	The network structure is visualized effectively and clearly, with appropriate labels and colors. Code is well-structured.	Visualization is mostly clear but has minor issues with labeling or presentation.	Visualization is unclear or poorly presented, making it difficult to understand the network structure.	Visualization is missing, poorly executed, or entirely incorrect.

Criteria	Excellent (4 points)	Good (3 points)	Fair (2 points)	Poor (1 point)
Clarity and Documentation	Code is well-documented, with clear comments explaining the logic and steps taken throughout the exercises.	Code has some documentation, but a few areas lack clarity or explanation.	Minimal documentation, making it difficult to follow the logic of the code.	No documentation provided; code is unclear and difficult to understand.
Creativity and Insight	Demonstrates exceptional creativity in implementing the exercises, providing additional insights or extensions.	Shows some creativity in implementation, but mostly follows provided guidelines.	Limited creativity; implementation is basic and does not explore additional insights.	No evidence of creativity; implementation is straightforward and lacks depth.