PROBABILISTIC REASONING

Labsheet-1

1. What is a Bayesian classifier?

A Bayesian classifier is a type of statistical classifier based on Bayes' Theorem, which provides a way to update the probability of a hypothesis as more evidence becomes available. It's widely used in machine learning for tasks such as classification, especiallywith text data. The most well-known example is the Naive Bayes classifier.

2. Why we use a Naive Bayes Classifier? Why it is called Naive?

A Naive Bayes classifier is used because it is simple, efficient, and effective for many classification tasks, especially with high-dimensional data like text. Despite its simplicity, it can perform well in various applications such as email spam detection,sentiment analysis, and medical diagnosis.

The classifier is termed "naive" because it makes a naive assumption: it assumes thatall features are conditionally independent given the class label.

3. What are the possible advantages in choosing Naive Bayes Classifier?

- Simple and Fast: Easy to implement and quick to train, even on large datasets.
- Effective with Small Data: Works well with limited training data.
- Performs Well with High-Dimensional Data: Great for text classification andother tasks with many features.
- Handles Multi-Class Problems: Naturally supports multi-class classification.
- Interpretable: Provides clear probability estimates for each class.
- Robust to Irrelevant Features: Can perform well even with noisy or irrelevant features.

4. Prepare a classification model using Naive Bayes for the given data.

5. Consider class Fish as Y1, class Animal as Y2, and class Bird as Y3 Compute P(Y1),P(Y2), P(Y3).

6. Consider the test sample X=(Slow, Rarely, No). Predict the class label for the test sample, using the Naive Bayes classifier. (Hint: Find P(Y1/X), P(Y2/X), and P(Y3/X)).

[* Executing Qns 4 to 6 in a single code *]

CODE

```
% Data preparation
Swim = {'Fast', 'Fast', 'Slow', 'Fast', 'No', 'No', 'No', 'Slow', 'Slow', 'Slow', 'No', 'Fast'};
Fly = {'No', 'No', 'No', 'No', 'Short', 'Short', 'Rarely', 'No', 'No', 'No', 'Long', 'No'};
Crawl = {'No', 'Yes', 'No', 'No', 'No', 'No', 'No', 'Yes', 'No', 'Yes', 'No', 'No'};
ClassLabel = {'Fish', 'Animal', 'Animal', 'Animal', 'Bird', 'Bird', 'Animal', 'Animal','Fish', 'Fish', 'Bird', 'Bird'};


% Convert categorical data into numeric
categories_swim = unique(Swim);
categories_fly = unique(Fly);
categories_crawl = unique(Crawl);
categories_class = unique(ClassLabel);


Swim_int = arrayfun(@(x) find(strcmp(x, categories_swim)), Swim); Fly_int
= arrayfun(@(x) find(strcmp(x, categories_fly)), Fly); Crawl_int =
arrayfun(@(x) find(strcmp(x, categories_crawl)), Crawl);
```

```
ClassLabel_int = arrayfun(@(x) find(strcmp(x, categories_class)), ClassLabel);



% Combine data into a matrix

data = [Swim_int, Fly_int, Crawl_int, ClassLabel_int];



% Step 2: Compute Prior Probabilities P(Y1), P(Y2), P(Y3)
```

```
num_classes = length(categories_class);

priors = histc(data(:, end), 1:num_classes) / size(data, 1);

disp('Prior Probabilities:');

for c = 1:num_classes

    fprintf('P(Y%d) = %.2f\n', c, priors(c));end



% Step 3: Calculate likelihoods P(X|Y) for each feature and class

num_features = size(data, 2) - 1;

likelihoods = cell(num_classes, num_features);for

c = 1:num_classes

    class_data = data(data(:, end) == c, 1:num_features);for

    f = 1:num_features

        feature_vals = unique(data(:, f));

        likelihoods{c, f} = histc(class_data(:, f), feature_vals) / size(class_data, 1);end

end



% Test sample X = (Slow, Rarely, No)

test_sample = [find(strcmp('Slow', categories_swim)), ...

               find(strcmp('Rarely', categories_fly)), ...

               find(strcmp('No', categories_crawl))];



% Step 4: Compute posterior probabilities P(Y|X) = P(X|Y)*P(Y)/P(X)

posteriors = zeros(num_classes, 1);
```

```
for c = 1:num_classes
    posterior = priors(c); for
    f = 1:num_features
        posterior = posterior * likelihoods{c, f}(test_sample(f));end
```

```
    posteriors(c) = posterior;
end


% Normalize posteriors to get P(Y1|X), P(Y2|X), P(Y3|X)
posteriors = posteriors / sum(posteriors);


% Display results disp('Posterior
Probabilities:');for c =
1:num_classes
    fprintf('P(Y%d|X) = %.2f\n', c, posteriors(c));end


% Predict the class label
[~, predicted_class_idx] = max(posteriors); predicted_class
= categories_class{predicted_class_idx};
fprintf('Predicted class for the test sample is: %s\n', predicted_class);
```

```
octave:12> source("mat_lab1.m")
Prior Probabilities:
P(Y1) = 0.42
P(Y2) = 0.33
P(Y3) = 0.25
Posterior Probabilities:
P(Y1|X) = 1.00
P(Y2|X) = 0.00
P(Y3|X) = 0.00
Predicted class for the test sample is: Animal
```

LAB-2

```
# Install the required library

!pip install pgmpy
```

```
any.whl.metadata (9.1 kB)

rkx in /usr/local/lib/python3.10/dist-packages (from pgmpy) (3.4.2)in
  /usr/local/lib/python3.10/dist-packages (from pgmpy) (1.26.4)

  in /usr/local/lib/python3.10/dist-packages (from pgmpy) (1.13.1)

t-learn in /usr/local/lib/python3.10/dist-packages (from pgmpy) (1.5.2)s in
/usr/local/lib/python3.10/dist-packages (from pgmpy) (2.2.2)

sing in /usr/local/lib/python3.10/dist-packages (from pgmpy) (3.2.0)

  in /usr/local/lib/python3.10/dist-packages (from pgmpy) (2.5.1+cu121) models in
/usr/local/lib/python3.10/dist-packages (from pgmpy) (0.14.4)in /usr/local/lib/python3.10/dist-
packages (from pgmpy) (4.66.6)

b  in /usr/local/lib/python3.10/dist-packages (from pgmpy) (1.4.2)

insum in /usr/local/lib/python3.10/dist-packages (from pgmpy) (3.4.0)st in
/usr/local/lib/python3.10/dist-packages (from pgmpy) (2.1.3)

e-generativeai in /usr/local/lib/python3.10/dist-packages (from pgmpy) (0.8.3)

e-ai-generativelanguage==0.6.10 in /usr/local/lib/python3.10/dist-packages (from google-generativeai->pgmpy) (0.6.10)e-api-core in
/usr/local/lib/python3.10/dist-packages (from google-generativeai->pgmpy) (2.19.2)

e-api-python-client in /usr/local/lib/python3.10/dist-packages (from google-generativeai->pgmpy) (2.151.0)e-auth>=2.15.0 in
/usr/local/lib/python3.10/dist-packages (from google-generativeai->pgmpy) (2.27.0)

buf in /usr/local/lib/python3.10/dist-packages (from google-generativeai->pgmpy) (4.25.5)tic in
/usr/local/lib/python3.10/dist-packages (from google-generativeai->pgmpy) (2.10.3)

g-extensions in /usr/local/lib/python3.10/dist-packages (from google-generativeai->pgmpy) (4.12.2)

-plus<2.0.0dev,>=1.22.3 in /usr/local/lib/python3.10/dist-packages (from google-ai-generativelanguage==0.6.10->google-generativean-dateutil>=2.8.2 in
/usr/local/lib/python3.10/dist-packages (from pandas->pgmpy) (2.8.2)

=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas->pgmpy) (2024.2) a>=2022.7 in
/usr/local/lib/python3.10/dist-packages (from pandas->pgmpy) (2024.2)

dpoolctl>=3.1.0 in /usr/local/lib/python3.10/dist-packages (from scikit-learn->pgmpy) (3.5.0)

>=0.5.6 in /usr/local/lib/python3.10/dist-packages (from statsmodels->pgmpy) (1.0.1) ging>=21.3 in
/usr/local/lib/python3.10/dist-packages (from statsmodels->pgmpy) (24.2)ock in /usr/local/lib/python3.10/dist-
packages (from torch->pgmpy) (3.16.1)

2 in /usr/local/lib/python3.10/dist-packages (from torch->pgmpy) (3.1.4)

c  in /usr/local/lib/python3.10/dist-packages (from torch->pgmpy) (2024.10.0)

==1.13.1 in /usr/local/lib/python3.10/dist-packages (from torch->pgmpy) (1.13.1)

h<1.4,>=1.1.0 in /usr/local/lib/python3.10/dist-packages (from sympy==1.13.1->torch->pgmpy) (1.3.0)a-nccl-cu12 in
/usr/local/lib/python3.10/dist-packages (from xgboost->pgmpy) (2.23.4)

eapis-common-protos<2.0.dev0,>=1.56.2 in /usr/local/lib/python3.10/dist-packages (from google-api-core->google-generativeai->pgmp

sts<3.0.0.dev0,>=2.18.0 in /usr/local/lib/python3.10/dist-packages (from google-api-core->google-generativeai->pgmpy) (2.32.3)tools<6.0,>=2.0.0 in
/usr/local/lib/python3.10/dist-packages (from google-auth>=2.15.0->google-generativeai->pgmpy) (5.5.0)

1-modules>=0.2.1 in /usr/local/lib/python3.10/dist-packages (from google-auth>=2.15.0->google-generativeai->pgmpy) (0.4.1)

,>=3.1.4 in /usr/local/lib/python3.10/dist-packages (from google-auth>=2.15.0->google-generativeai->pgmpy) (4.9)

1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.8.2->pandas->pgmpy) (1.16.0)

ib2<1.dev0,>=0.19.0 in /usr/local/lib/python3.10/dist-packages (from google-api-python-client->google-generativeai->pgmpy) (0.22.e-auth-httplib2<1.0.0,>=0.2.0 in
/usr/local/lib/python3.10/dist-packages (from google-api-python-client->google-generativeai->pgmmplate<5,>=3.0.1 in /usr/local/lib/python3.10/dist-packages (from google-
api-python-client->google-generativeai->pgmpy) (4.1.1) pSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from jinja2->torch->pgmpy) (3.0.2)

ated-types>=0.6.0 in /usr/local/lib/python3.10/dist-packages (from pydantic->google-generativeai->pgmpy) (0.7.0)tic-core==2.27.1 in
/usr/local/lib/python3.10/dist-packages (from pydantic->google-generativeai->pgmpy) (2.27.1)
```

o<2.0dev,>=1.33.2 in /usr/local/lib/python3.10/dist-packages (from google-api-core[grpc]!=2.0.*,!=2.1.*,!=2.10.*,!=2.2.*,!=2.3.*, o-status<2.0.dev0,>=1.33.2 in /usr/local/lib/python3.10/dist-packages (from google-api-core[grpc]!=2.0.*,!=2.1.*,!=2.10.*,!=2.2.* 1<0.7.0,>=0.4.6 in /usr/local/lib/python3.10/dist-packages (from pyasn1-modules>=0.2.1->google-auth>=2.15.0->google-generativeai- et-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests<3.0.0.dev0,>=2.18.0->google-api-core->google-genera 4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests<3.0.0.dev0,>=2.18.0->google-api-core->google-generativeai->pgmp b3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests<3.0.0.dev0,>=2.18.0->google-api-core->google-generativeai fi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests<3.0.0.dev0,>=2.18.0->google-api-core->google-generativeaiy.whl (2.0 MB)

━━━━━━━━━━ 2.0/2.0 MB 20.2 MB/s eta 0:00:00

Install pgmpy and supporting libraries

!pip install pgmpy networkx matplotlib

Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-packages (3.8.0)

Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (from pgmpy) (1.26.4)Requirement already satisfied: scipy in /usr/local/lib/python3.10/dist-packages (from pgmpy) (1.13.1)

Requirement already satisfied: scikit-learn in /usr/local/lib/python3.10/dist-packages (from pgmpy) (1.5.2)Requirement already satisfied: pandas in /usr/local/lib/python3.10/dist-packages (from pgmpy) (2.2.2)

Requirement already satisfied: pyparsing in /usr/local/lib/python3.10/dist-packages (from pgmpy) (3.2.0)

Requirement already satisfied: torch in /usr/local/lib/python3.10/dist-packages (from pgmpy) (2.5.1+cu121) Requirement already satisfied: statsmodels in /usr/local/lib/python3.10/dist-packages (from pgmpy) (0.14.4)Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (from pgmpy) (4.66.6)

Requirement already satisfied: joblib in /usr/local/lib/python3.10/dist-packages (from pgmpy) (1.4.2)

Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (1.3.1)Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (0.12.1)

Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (4.55.1)Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (1.4.7) Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (24.2)

Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (11.0.0)

Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (2.8.2)

Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.7->matplotlib) (1.16Requirement already satisfied: google-ai-generativelanguage==0.6.10 in /usr/local/lib/python3.10/dist-packages (from google-geneRequirement already satisfied: google-api-core in /usr/local/lib/python3.10/dist-packages (from google-generativeai->pgmpy) (2.1Requirement already satisfied: google-api-python-client in /usr/local/lib/python3.10/dist-packages (from google-generativeai->pgRequirement already satisfied: google-auth>=2.15.0 in /usr/local/lib/python3.10/dist-packages (from google-generativeai->pgmpy) Requirement already satisfied: protobuf in /usr/local/lib/python3.10/dist-packages (from google-generativeai->pgmpy) (4.25.5)

Requirement already satisfied: pydantic in /usr/local/lib/python3.10/dist-packages (from google-generativeai->pgmpy) (2.10.3)

Requirement already satisfied: typing-extensions in /usr/local/lib/python3.10/dist-packages (from google-generativeai->pgmpy) (4 Requirement already satisfied: proto-plus<2.0.0dev,>=1.22.3 in /usr/local/lib/python3.10/dist-packages (from google-ai-generativ Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas->pgmpy) (2024.2)

Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.10/dist-packages (from pandas->pgmpy) (2024.2)

Requirement already satisfied: threadpoolctl>=3.1.0 in /usr/local/lib/python3.10/dist-packages (from scikit-learn->pgmpy) (3.5.0Requirement already satisfied: patsy>=0.5.6 in /usr/local/lib/python3.10/dist-packages (from statsmodels->pgmpy) (1.0.1)

Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-packages (from torch->pgmpy) (3.16.1)Requirement already satisfied: jinja2 in /usr/local/lib/python3.10/dist-packages (from torch->pgmpy) (3.1.4)

Requirement already satisfied: fsspec in /usr/local/lib/python3.10/dist-packages (from torch->pgmpy) (2024.10.0)

Requirement already satisfied: sympy==1.13.1 in /usr/local/lib/python3.10/dist-packages (from torch->pgmpy) (1.13.1)

Requirement already satisfied: mpmath<1.4,>=1.1.0 in /usr/local/lib/python3.10/dist-packages (from sympy==1.13.1->torch->pgmpy)Requirement already satisfied: nvidia-nccl-cu12 in /usr/local/lib/python3.10/dist-packages (from xgboost->pgmpy) (2.23.4)

Requirement already satisfied: googleapis-common-protos<2.0.dev0,>=1.56.2 in /usr/local/lib/python3.10/dist-packages (from googl Requirement already satisfied: requests<3.0.0.dev0,>=2.18.0 in /usr/local/lib/python3.10/dist-packages (from google-api-core->go Requirement already satisfied: cachetools<6.0,>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from google-auth>=2.15.0->goog Requirement already satisfied: pyasn1-modules>=0.2.1 in /usr/local/lib/python3.10/dist-packages (from google-auth>=2.15.0->googl Requirement already satisfied: rsa<5,>=3.1.4 in /usr/local/lib/python3.10/dist-packages (from google-auth>=2.15.0->google-genera Requirement already satisfied: httplib2<1.dev0,>=0.19.0 in /usr/local/lib/python3.10/dist-packages (from google-api-python-clien Requirement already satisfied: google-auth-httplib2<1.0.0,>=0.2.0 in /usr/local/lib/python3.10/dist-packages (from google-api-pyRequirement already satisfied: uritemplate<5,>=3.0.1 in /usr/local/lib/python3.10/dist-packages (from google-api-python-client->Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from jinja2->torch->pgmpy) (3.0.2)

Requirement already satisfied: annotated-types>=0.6.0 in /usr/local/lib/python3.10/dist-packages (from pydantic->google-generati Requirement already satisfied: pydantic-core==2.27.1 in /usr/local/lib/python3.10/dist-packages (from pydantic->google-generativ Requirement already satisfied: grpcio<2.0dev,>=1.33.2 in /usr/local/lib/python3.10/dist-packages (from google-api-core[grpc]!=2. Requirement already satisfied: grpcio-status<2.0.dev0,>=1.33.2 in /usr/local/lib/python3.10/dist-packages (from google-api-core[Requirement already satisfied: pyasn1<0.7.0,>=0.4.6 in /usr/local/lib/python3.10/dist-packages (from pyasn1-modules>=0.2.1->goog Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests<3.0.0.dev0,>=2Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests<3.0.0.dev0,>=2.18.0->googl Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests<3.0.0.dev0,>=2.18.0-Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests<3.0.0.dev0,>=2.18.0-

```
from pgmpy.models import BayesianNetwork

# Step 1: Define the Bayesian Network
structuremodel = BayesianNetwork([

    ('Diff', 'Grade'), # Difficulty influences Grade
    ('Intel', 'Grade'), # Intelligence influences Grade
    ('Grade', 'Letter'),# Grade influences Letter

    ('Intel', 'SAT')])   # Intelligence influences SAT

# Step 2: Define the CPDs (Conditional Probability

Distributions)from pgmpy.factors.discrete import TabularCPD


# CPD for Difficulty
```

```
cpd_diff = TabularCPD(variable='Diff', variable_card=2,
                      values=[[0.6], [0.4]],
                      state_names={'Diff': ['Easy', 'Hard']})

# CPD for Intelligence
cpd_intel = TabularCPD(variable='Intel', variable_card=2,
                       values=[[0.7], [0.3]],
                       state_names={'Intel': ['Dumb', 'Intelligent']})

# CPD for Grade
cpd_grade = TabularCPD(variable='Grade', variable_card=3,
                       values=[
                           [0.3, 0.05, 0.9, 0.5],   # Grade A (G=0)
                           [0.4, 0.25, 0.08, 0.3], # Grade B (G=1)
                           [0.3, 0.7,   0.02, 0.2]   # Grade C (G=2)
                       ],
                       evidence=['Intel', 'Diff'], evidence_card=[2, 2],
                       state_names={'Grade': ['A', 'B', 'C'], 'Intel': ['Dumb', 'Intelligent'], 'Diff': ['Easy', 'Hard']})

# CPD for SAT
cpd_sat = TabularCPD(variable='SAT', variable_card=2,
                     values=[
                         [0.95, 0.2],   # SAT Bad (S=0)
                         [0.05, 0.8]     # SAT Good (S=1)
```

```
                        ],
                        evidence=['Intel'], evidence_card=[2],
                        state_names={'SAT': ['Bad', 'Good'], 'Intel': ['Dumb', 'Intelligent']})


# CPD for Letter
cpd_letter = TabularCPD(variable='Letter', variable_card=2,
                            values=[
                                [0.1, 0.4, 0.99],   # Letter Bad (L=0)
                                [0.9, 0.6, 0.01]      # Letter Good (L=1)
                            ],
                            evidence=['Grade'], evidence_card=[3],
                            state_names={'Letter': ['Bad', 'Good'], 'Grade': ['A', 'B', 'C']})
# Add CPDs to the
model


model.add_cpds(cpd_diff, cpd_intel, cpd_grade, cpd_sat, cpd_letter)


# Validate the model
assert model.check_model()


print("CPDs:")
print(cpd_diff)
print(cpd_intel
)
print(cpd_grad
e)
print(cpd_sat)
print(cpd_letter)
```

```
CPDs:
+------------+-----+
| Diff(Easy) | 0.6 |
+------------+-----+
| Diff(Hard) | 0.4 |
```

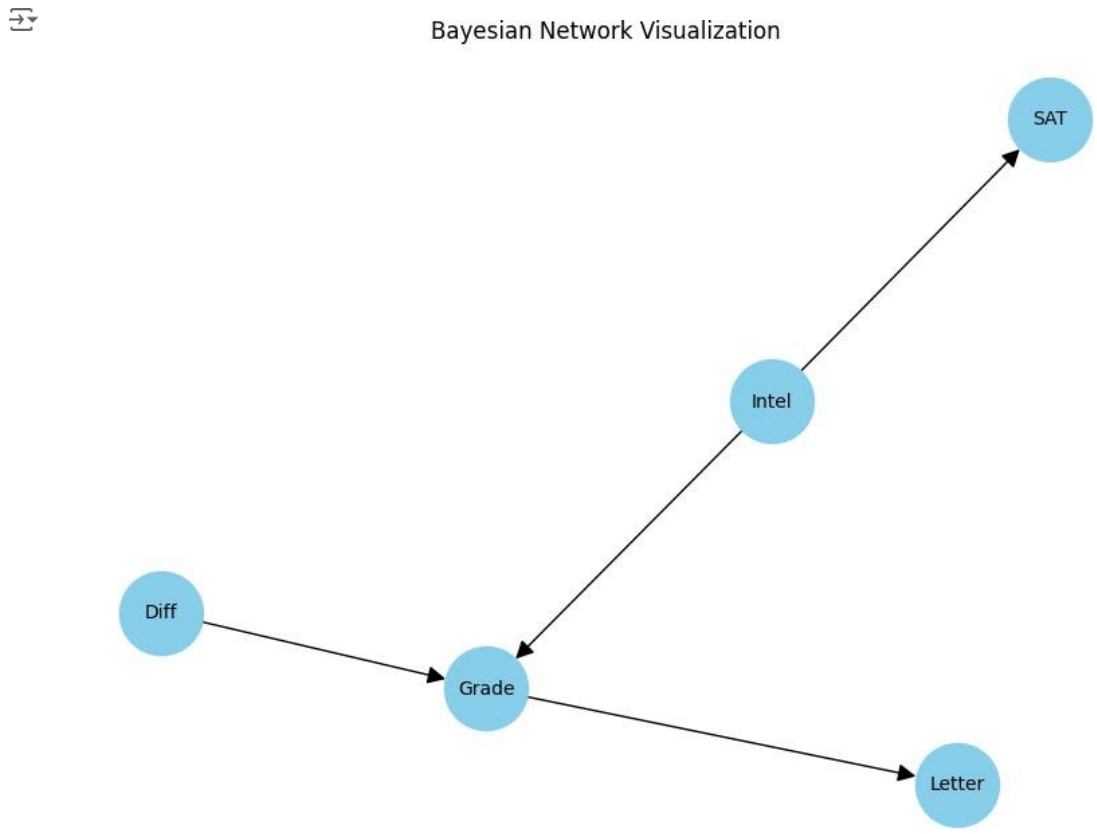
```
+------------+-----+
+--------------------+-----+
| Intel(Dumb)                    | 0.7 |
+--------------------+-----+
| Intel(Intelligent) | 0.3 |
+--------------------+-----+
+----------+-------------+-------------+--------------------+--------------------+
| Intel            | Intel(Dumb) | Intel(Dumb) | Intel(Intelligent) | Intel(Intelligent) |
+----------+-------------+-------------+--------------------+--------------------+
| Diff             | Diff(Easy)   | Diff(Hard)   | Diff(Easy)                          | Diff(Hard)                   |
+----------+-------------+-------------+--------------------+--------------------+
| Grade(A) | 0.3                      | 0.05                | 0.9                          | 0.5                          |
+----------+-------------+-------------+--------------------+--------------------+
| Grade(B) | 0.4                      | 0.25                | 0.08                         | 0.3                          |
+----------+-------------+-------------+--------------------+--------------------+
| Grade(C) | 0.3                      | 0.7                 | 0.02                         | 0.2                          |
+----------+-------------+-------------+--------------------+--------------------+
+-----------+-------------+--------------------+
| Intel               | Intel(Dumb) | Intel(Intelligent) |
+-----------+-------------+--------------------+
| SAT(Bad)   | 0.95                    | 0.2                          |
+-----------+-------------+--------------------+
| SAT(Good) | 0.05                    | 0.8                          |
+-----------+-------------+--------------------+
+--------------+----------+----------+----------+
| Grade                 | Grade(A) | Grade(B) | Grade(C) |
+--------------+----------+----------+----------+
| Letter(Bad)   | 0.1                    | 0.4             | 0.99            |
+--------------+----------+----------+----------+
| Letter(Good) | 0.9                    | 0.6             | 0.01            |
+--------------+----------+----------+----------+
```

```
# Step 3: Local independencies and active trail nodes# Local
Independencies

print("Local Independencies:")

print(model.local_independencies('Diff'))
print(model.local_independencies('Intel'))
print(model.local_independencies('Grade'))
print(model.local_independencies('SAT'))
```

```
print(model.local_independencies('Letter'))


# Active trail nodes for 'Diff'

print("\nActive trail nodes for 'Diff':")
print(model.active_trail_nodes('Diff'))
```

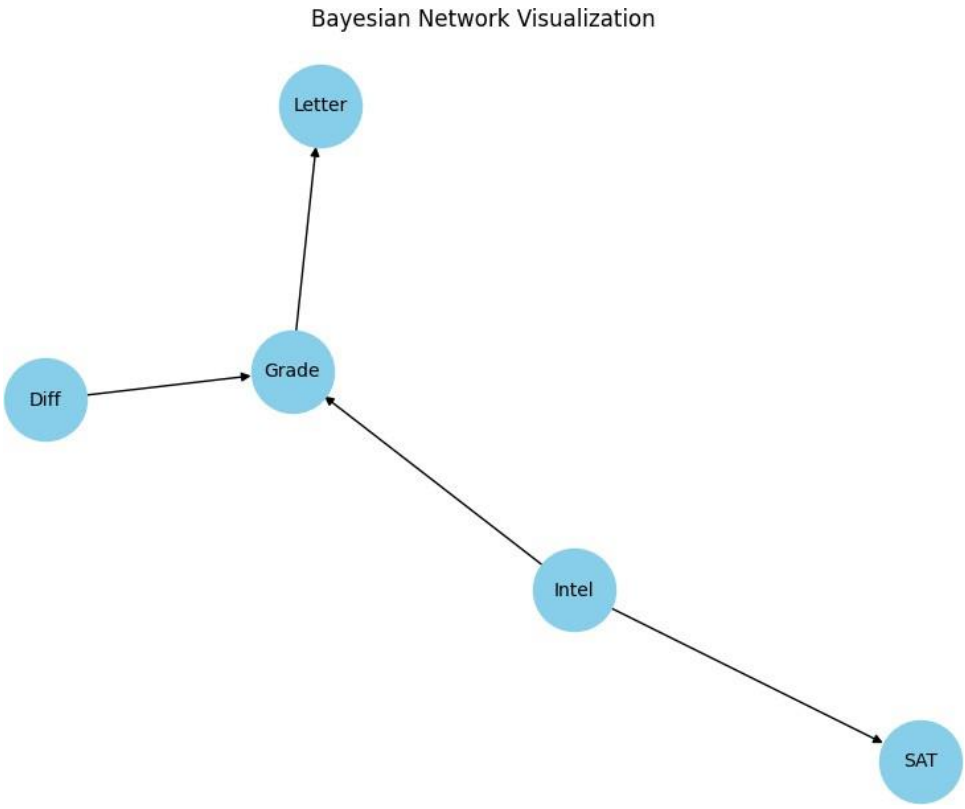
```
Local Independencies:(Diff ⊥
SAT, Intel) (Intel ⊥ Diff)

(Grade ⊥ SAT | Diff, Intel)

(SAT ⊥ Diff, Grade, Letter | Intel)(Letter ⊥ SAT,
Diff, Intel | Grade)


Active trail nodes for 'Diff':

{'Diff': {'Diff', 'Grade', 'Letter'}}
```

```
# Step 4: Visualize the Bayesian Networkimport
networkx as nx

import matplotlib.pyplot as plt



# Manually convert the BayesianNetwork into a networkx graph graph =
nx.DiGraph() # Directed graph for Bayesian networksfor edge in model.edges():

    graph.add_edge(edge[0], edge[1])



# Visualize the graph using networkx
plt.figure(figsize=(8, 6))

nx.draw(graph, with_labels=True, node_color="skyblue", font_size=10, node_size=2000, edge_color="black", arrowsize=20)plt.title("Bayesian Network
Visualization")

plt.show()
```




```
import networkx as nx

import matplotlib.pyplot as plt



# Create a directed graph using networkxgraph =
nx.DiGraph()



# Add nodes and edges from the Bayesian Network
graph.add_nodes_from(model.nodes())

graph.add_edges_from(model.edges())
```

```
# Visualize the graph using networkx
plt.figure(figsize=(8, 6))

nx.draw(graph, with_labels=True, node_color="skyblue", font_size=10, node_size=2000, edge_color="black")plt.title("Bayesian Network
Visualization")

plt.show()
```


Bayesian Network Visualization
Letter
Diff
Grade
Intel
SAT

LAB-3

```
# Install the pgmpy library

!pip install pgmpy
```

```
Collecting pgmpy
  Downloading pgmpy-0.1.26-py3-none-any.whl.metadata (9.1 kB)
Requirement already satisfied: networkx in /usr/local/lib/python3.10/dist-packages (from pgmpy) (3.4.2)Requirement already satisfied:
numpy in /usr/local/lib/python3.10/dist-packages (from pgmpy) (1.26.4)
Requirement already satisfied: scipy in /usr/local/lib/python3.10/dist-packages (from pgmpy) (1.13.1)
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.10/dist-packages (from pgmpy) (1.5.2)Requirement already satisfied: pandas in
/usr/local/lib/python3.10/dist-packages (from pgmpy) (2.2.2)
Requirement already satisfied: pyparsing in /usr/local/lib/python3.10/dist-packages (from pgmpy) (3.2.0)
Requirement already satisfied: torch in /usr/local/lib/python3.10/dist-packages (from pgmpy) (2.5.1+cu121) Requirement already satisfied:
statsmodels in /usr/local/lib/python3.10/dist-packages (from pgmpy) (0.14.4)Requirement already satisfied: tqdm in
/usr/local/lib/python3.10/dist-packages (from pgmpy) (4.66.6)
Requirement already satisfied: joblib in /usr/local/lib/python3.10/dist-packages (from pgmpy) (1.4.2)
Requirement already satisfied: opt-einsum in /usr/local/lib/python3.10/dist-packages (from pgmpy) (3.4.0)Requirement already satisfied:
xgboost in /usr/local/lib/python3.10/dist-packages (from pgmpy) (2.1.3)
Requirement already satisfied: google-generativeai in /usr/local/lib/python3.10/dist-packages (from pgmpy) (0.8.3)
Requirement already satisfied: google-ai-generativelanguage==0.6.10 in /usr/local/lib/python3.10/dist-packages (from google-geneRequirement already satisfied: google-api-
core in /usr/local/lib/python3.10/dist-packages (from google-generativeai->pgmpy) (2.1Requirement already satisfied: google-api-python-client in
/usr/local/lib/python3.10/dist-packages (from google-generativeai->pgRequirement already satisfied: google-auth>=2.15.0 in /usr/local/lib/python3.10/dist-packages (from
google-generativeai->pgmpy) Requirement already satisfied: protobuf in /usr/local/lib/python3.10/dist-packages (from google-generativeai->pgmpy) (4.25.5)
Requirement already satisfied: pydantic in /usr/local/lib/python3.10/dist-packages (from google-generativeai->pgmpy) (2.10.3)
Requirement already satisfied: typing-extensions in /usr/local/lib/python3.10/dist-packages (from google-generativeai->pgmpy) (4 Requirement already satisfied: proto-
plus<2.0.0dev,>=1.22.3 in /usr/local/lib/python3.10/dist-packages (from google-ai-generativ Requirement already satisfied: python-dateutil>=2.8.2 in
/usr/local/lib/python3.10/dist-packages (from pandas->pgmpy) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas->pgmpy) (2024.2)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.10/dist-packages (from pandas->pgmpy) (2024.2)
Requirement already satisfied: threadpoolctl>=3.1.0 in /usr/local/lib/python3.10/dist-packages (from scikit-learn->pgmpy) (3.5.0Requirement already satisfied: patsy>=0.5.6 in
/usr/local/lib/python3.10/dist-packages (from statsmodels->pgmpy) (1.0.1)
Requirement already satisfied: packaging>=21.3 in /usr/local/lib/python3.10/dist-packages (from statsmodels->pgmpy) (24.2)Requirement already satisfied: filelock in
/usr/local/lib/python3.10/dist-packages (from torch->pgmpy) (3.16.1)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.10/dist-packages (from torch->pgmpy) (3.1.4)
Requirement already satisfied: fsspec in /usr/local/lib/python3.10/dist-packages (from torch->pgmpy) (2024.10.0)
Requirement already satisfied: sympy==1.13.1 in /usr/local/lib/python3.10/dist-packages (from torch->pgmpy) (1.13.1)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in /usr/local/lib/python3.10/dist-packages (from sympy==1.13.1->torch->pgmpy)Requirement already satisfied: nvidia-
nccl-cu12 in /usr/local/lib/python3.10/dist-packages (from xgboost->pgmpy) (2.23.4)
Requirement already satisfied: googleapis-common-protos<2.0.dev0,>=1.56.2 in /usr/local/lib/python3.10/dist-packages (from googl
Requirement already satisfied: requests<3.0.0.dev0,>=2.18.0 in /usr/local/lib/python3.10/dist-packages (from google-api-core->goRequirement already satisfied:
cachetools<6.0,>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from google-auth>=2.15.0->googRequirement already satisfied: pyasn1-modules>=0.2.1 in
/usr/local/lib/python3.10/dist-packages (from google-auth>=2.15.0->googlRequirement already satisfied: rsa<5,>=3.1.4 in /usr/local/lib/python3.10/dist-packages (from
google-auth>=2.15.0->google-generaRequirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.8.2->pandas->pgmpy)
Requirement already satisfied: httplib2<1.dev0,>=0.19.0 in /usr/local/lib/python3.10/dist-packages (from google-api-python-clienRequirement already satisfied: google-auth-
httplib2<1.0.0,>=0.2.0 in /usr/local/lib/python3.10/dist-packages (from google-api-pyRequirement already satisfied: uritemplate<5,>=3.0.1 in /usr/local/lib/python3.10/dist-
packages (from google-api-python-client->Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from jinja2->torch->pgmpy) (3.0.2)
```

```
Requirement already satisfied: annotated-types>=0.6.0 in /usr/local/lib/python3.10/dist-packages (from pydantic->google-generatiRequirement already satisfied: pydantic-core==2.27.1 in /usr/local/lib/python3.10/dist-packages (from pydantic->google-generativ Requirement already satisfied: grpcio<2.0dev,>=1.33.2 in /usr/local/lib/python3.10/dist-packages (from google-api-core[grpc]!=2. Requirement already satisfied: grpcio-status<2.0.dev0,>=1.33.2 in /usr/local/lib/python3.10/dist-packages (from google-api-core[ Requirement already satisfied: pyasn1<0.7.0,>=0.4.6 in /usr/local/lib/python3.10/dist-packages (from pyasn1-modules>=0.2.1->goog Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests<3.0.0.dev0,>=2Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests<3.0.0.dev0,>=2.18.0->googl Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests<3.0.0.dev0,>=2.18.0-Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests<3.0.0.dev0,>=2.18.0-Downloading pgmpy-0.1.26-py3-none-any.whl (2.0 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
2.0/2.0 MB 17.6 MB/s eta 0:00:00Installing collected packages: pgmpy
```

```
# Import necessary modules

from pgmpy.models import MarkovNetwork

from pgmpy.factors.discrete import DiscreteFactorfrom
pgmpy.inference import VariableElimination



# Step 1: Define the Markov Network# Create
an empty Markov Network

model = MarkovNetwork()



# Add edges as per the network structure from the table

model.add_edges_from([('A', 'B'), ('B', 'C'), ('C', 'D'), ('D', 'A')])



# Step 2: Add the Factors# Factor
ϕ(A, B)

phi_AB = DiscreteFactor(
    variables=['A', 'B'],
```

```
        cardinality=[2, 2], # Binary variablesvalues=[

                [30, 5], # A=0, B=0 | A=0, B=1[1, 10]  #
                A=1, B=0 | A=1, B=1

        ]

)



# Factor ϕ(B, C)

phi_BC = DiscreteFactor(
        variables=['B', 'C'],
        cardinality=[2, 2], values=[

                [100, 1], # B=0, C=0 | B=0, C=1[1, 100]
                                # B=1, C=0 | B=1, C=1

        ]

)



# Factor ϕ(C, D)

phi_CD = DiscreteFactor(
        variables=['C', 'D'],
        cardinality=[2, 2], values=[

                [1, 100], # C=0, D=0 | C=0, D=1[100, 1]
                                # C=1, D=0 | C=1, D=1

        ]

)



# Factor ϕ(D, A)

phi_DA = DiscreteFactor(
        variables=['D', 'A'],
        cardinality=[2, 2], values=[

                [100, 1], # D=0, A=0 | D=0, A=1[1, 100]
                                # D=1, A=0 | D=1, A=1

        ]

)



# Add these factors to the model

model.add_factors(phi_AB, phi_BC, phi_CD, phi_DA)



# Step 3: Perform inference

inference = VariableElimination(model)



# Example query: MAP estimation for variable 'C' given evidence A=0 and B=1result =
inference.map_query(variables=['C'], evidence={'A': 0, 'B': 1})

print("\nMAP Query result (C given A=0, B=1):")print(result)



# Example query: Probability of 'C' given evidence

prob_result = inference.query(variables=['C'], evidence={'A': 0, 'B': 1})print("\nProbability
distribution of 'C' given A=0, B=1:")
```

```
print(prob_result)
```

```
Eliminating: D: 100%                                    1/1 [00:00<00:00, 63.14it/s]

MAP Query result (C given A=0, B=1):

{'C': 1}


Probability distribution of 'C' given A=0, B=1:

+------+-------------+
| C    |      phi(C) |
+======+=============+
| C(0) |   1000.0000 |
+------+-------------+
| C(1) | 5000500.0000 |
+------+-------------+
```