AI ASSIGNMENT 3 REPORT – ANTONIO MANUEL LUQUE MOLINA

Introduction

This report presents the construction and analysis of Bayesian Networks (BNs) to model relationships between various weather variables. Using both exact and approximate inference methods, we explore how different network structures influence the predictive capabilities of the BNs. Each section of this report outlines the methods used, discusses the results obtained, and provides insights into the implications of these results.

Task 1: Bayesian Network for Weather Prediction.

1.1 Bayesian Network Construction

Methodology:
A Bayesian Network was constructed to model relationships specified in Figure 1:

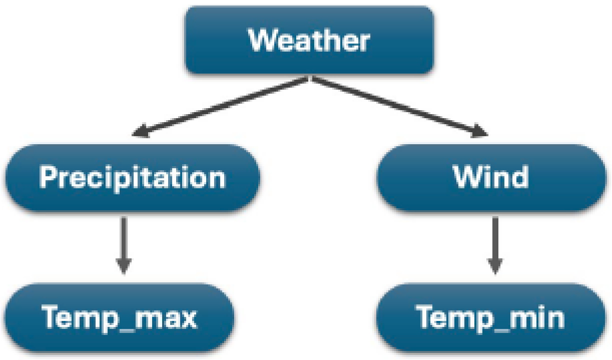


Fig1: Sample Hierarchy in consideration for the given data

We developed the following code in Python in order to construct and configure the Bayesian Network that models relationships among weather-related variables specified in the figure:

1. **Network Structure Definition and State Specification**:

```
# Define the hierarchy
modelfig1 = BayesianNetwork([
    ('weather', 'precipitation'), ('weather', 'wind'),
    ('precipitation', 'temp_max'),
    ('wind', 'temp_min')
])

# And, the states for each variables
weather_states = ['drizzle', 'rain', 'sun', 'snow', 'fog']
precipitation_states = ['low', 'mid', 'high']
wind_states = ['low', 'mid', 'high']
temp_max_states = ['low', 'mid', 'high']
temp_min_states = ['low', 'mid', 'high']
```

2. **Probability Calculation**:

- **Marginal Probabilities**: Directly calculated for the **weather** node from data, reflecting the frequency of each weather state.

```
weather_marginal = (df['weather'].value_counts()/len(df['weather'])).round(3)
weather_marginal = np.array([[value] for value in weather_marginal])
```

- **Conditional Probabilities**: Calculated for dependent variables like precipitation and temperatures based on their parent node states. This includes handling missing data combinations by assigning equal probabilities where data is insufficient.

```
# Joint Propabilities
# Create dict where key=parent, value=child
var_dict = {'weather': ['precipitation', 'wind'],
            'precipitation': ['temp_max'],
            'wind': ['temp_min']
            }
```

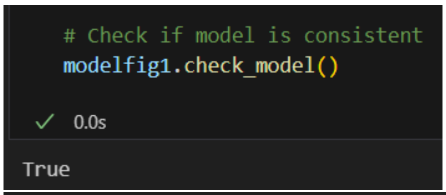
3. **CPD Integration**: Constructs and attaches Conditional Probability Distributions (CPDs) to each node in the network. These CPDs encapsulate the probability calculations, linking theoretical network structure with empirical data.

```
cpd_lst = []
for key, value in var_dict.items():
    length = len(value)
    for i in range(length):
        value_given_key =
df.groupby(key)[value[i]].value_counts(normalize=True).sort_index()
        cpd = value_given_key.unstack(fill_value=0).to_numpy().T
        cpd_lst.append(cpd)
# Note that we get 3 Nan values in the above conditional distributions. This is because
one of the type of precipitation (low) did not contain any relation with temp_max.
# Therefore, normalization, does not produce the intended result.
# To mitigate this, we replace Nan with the equal probability within the three values,
i.e., 0.33
cpd_lst[2][:,0] = .33
cpd_lst
```

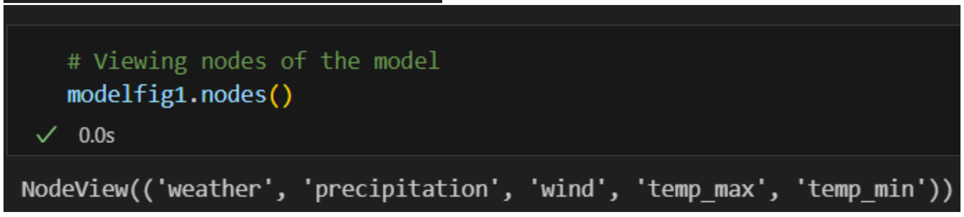
```
# Creating tabular conditional probability distribution
weather_cpd = TabularCPD(variable='weather', variable_card=5, values=weather_marginal,
state_names={'weather': weather_states})

precipitation_cpd = TabularCPD(variable='precipitation', variable_card=3,
evidence=['weather'], evidence_card=[5],
                               values=cpd_lst[0], state_names={'precipitation':
precipitation_states, 'weather': weather_states})

wind_cpd = TabularCPD(variable='wind', variable_card=3, evidence=['weather'],
evidence_card=[5],
                               values=cpd_lst[1], state_names={'wind': wind_states,
'weather': weather_states})
```

```
temp_max_cpd = TabularCPD(variable='temp_max', variable_card=3,
evidence=['precipitation'], evidence_card=[3],
                              values=cpd_lst[2], state_names={'temp_max':
temp_max_states, 'precipitation': precipitation_states})

temp_min_cpd = TabularCPD(variable='temp_min', variable_card=3, evidence=['wind'],
evidence_card=[3],
                              values=cpd_lst[3], state_names={'temp_min':
temp_min_states, 'wind': wind_states})

# Add CPDs and factors to the model
modelfig1.add_cpds(weather_cpd, precipitation_cpd, wind_cpd, temp_max_cpd, temp_min_cpd)
```

4. **Model Validation and Analysis**: Ensures the BN's consistency and correctness through checks and enables exploration of network properties like node independencies and conditional relationships



```
# Check if model is consistent
modelfig1.check_model()
✓ 0.0s
True
```



```
# Viewing nodes of the model
modelfig1.nodes()
✓ 0.0s
NodeView(('weather', 'precipitation', 'wind', 'temp_max', 'temp_min'))
```

More analysis can be seen in the code.

Relevance:
The construction of this BN is fundamental as it provides a graphical model that can be used to compute probabilities of interest, exploring how different weather variables interact.

1.2 Exact Inference using Variable Elimination

Exact inference using Variable Elimination is a computational method in probabilistic graphical models, like Bayesian networks. It systematically calculates the exact probability distribution for a subset of variables by summing out or eliminating the non-relevant variables from the joint probability distribution. This method follows a sequence of operations on probability tables, effectively reducing the problem size and focusing only on the variables of interest, thus providing precise results.

Methodology:
Variable Elimination (VE) was applied to perform exact inference on the network, allowing for the computation of marginal probabilities and conditional probabilities based on the BN structure.

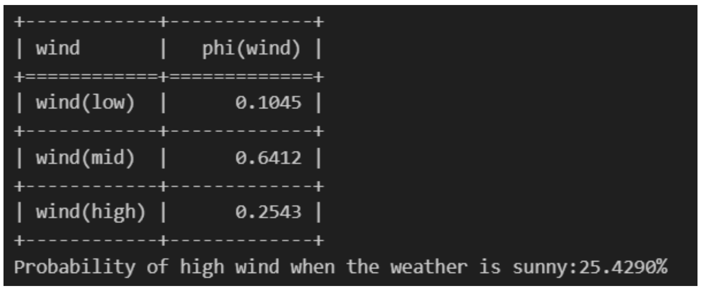
We used this function with all the tasks:

query(*variables*, *evidence=None*, *virtual_evidence=None*, *elimination_order='greedy'*, *joint=True*, *s how_progress=True*)

from the *class* pgmpy.inference.ExactInference.**VariableElimination**(*model*)

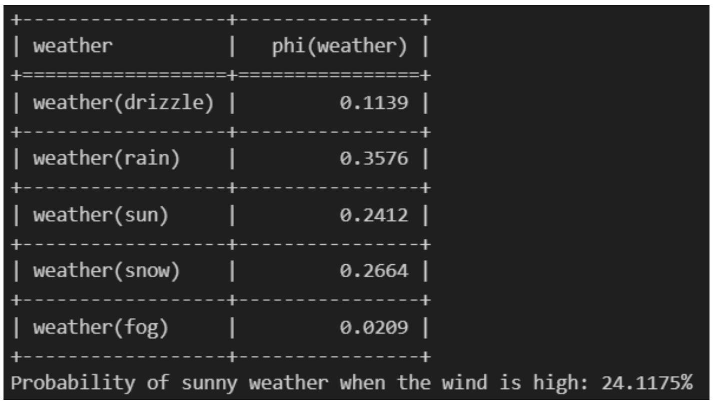
Results Analysis:

- **1.2.1 (a)** Probability of high wind when the weather is sunny was calculated, providing insights into how sunny weather can influence wind conditions:



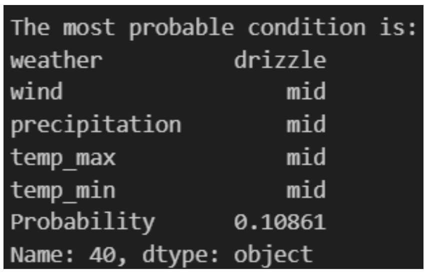
```
+------------+------------+
| wind       |   phi(wind) |
+============+============+
| wind(low)  |     0.1045 |
+------------+------------+
| wind(mid)  |     0.6412 |
+------------+------------+
| wind(high) |     0.2543 |
+------------+------------+
Probability of high wind when the weather is sunny:25.4290%
```

- **1.2.1 (b)** Probability of sunny weather when the wind is high was computed to understand the reverse influence of wind on weather conditions:



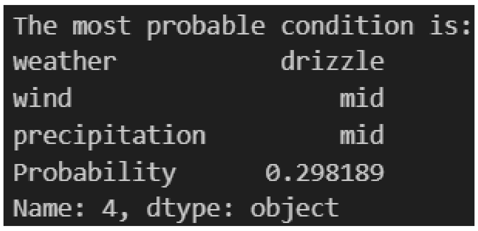
```
+------------------+----------------+
| weather          |   phi(weather) |
+==================+================+
| weather(drizzle) |         0.1139 |
+------------------+----------------+
| weather(rain)    |         0.3576 |
+------------------+----------------+
| weather(sun)     |         0.2412 |
+------------------+----------------+
| weather(snow)    |         0.2664 |
+------------------+----------------+
| weather(fog)     |         0.0209 |
+------------------+----------------+
Probability of sunny weather when the wind is high: 24.1175%
```

- **1.2.2 (a)** All possible joint probabilities were computed, identifying the most probable overall weather condition:



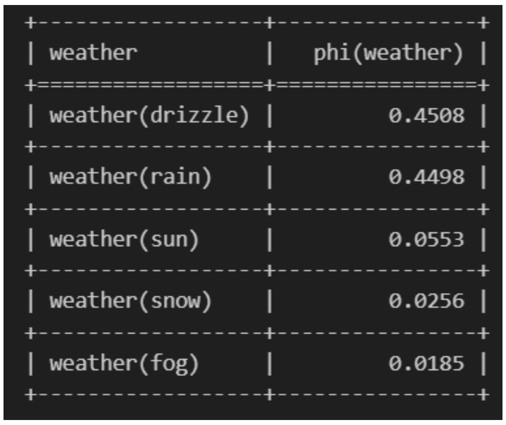
```
The most probable condition is:
weather          drizzle
wind                 mid
precipitation        mid
temp_max             mid
temp_min             mid
Probability      0.10861
Name: 40, dtype: object
```

- **1.2.2 (b)** Determination of the most probable condition for precipitation, wind, and weather combined:



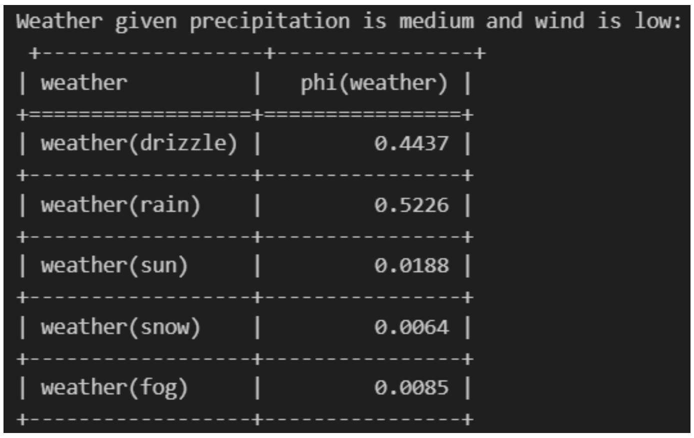
```
The most probable condition is:
weather           drizzle
wind                  mid
precipitation         mid
Probability      0.298189
Name: 4, dtype: object
```

- **1.2.3** The probabilities associated with each weather condition given medium precipitation were evaluated:

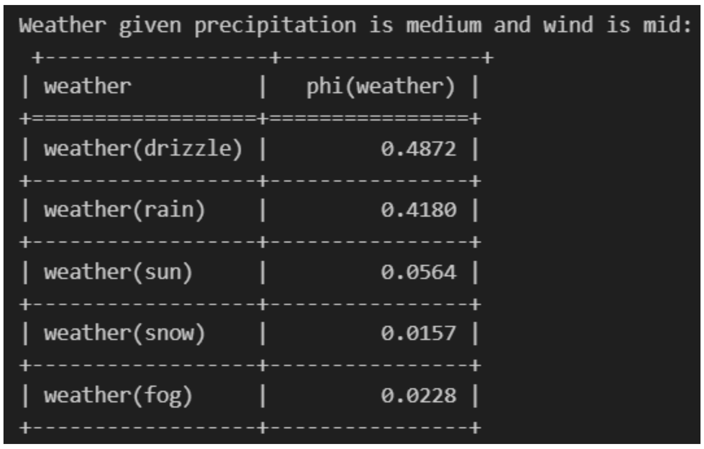


```
+------------------+----------------+
| weather          |   phi(weather) |
+==================+================+
| weather(drizzle) |         0.4508 |
+------------------+----------------+
| weather(rain)    |         0.4498 |
+------------------+----------------+
| weather(sun)     |         0.0553 |
+------------------+----------------+
| weather(snow)    |         0.0256 |
+------------------+----------------+
| weather(fog)     |         0.0185 |
+------------------+----------------+
```

- **1.2.4** Analysis of how the addition of wind conditions (low or medium) affects the probabilities calculated in 1.2.3.



```
Weather given precipitation is medium and wind is low:
 +------------------+----------------+
| weather          |   phi(weather) |
+==================+================+
| weather(drizzle) |         0.4437 |
+------------------+----------------+
| weather(rain)    |         0.5226 |
+------------------+----------------+
| weather(sun)     |         0.0188 |
+------------------+----------------+
| weather(snow)    |         0.0064 |
+------------------+----------------+
| weather(fog)     |         0.0085 |
+------------------+----------------+
```



```
Weather given precipitation is medium and wind is mid:
 +------------------+----------------+
| weather          |   phi(weather) |
+==================+================+
| weather(drizzle) |         0.4872 |
+------------------+----------------+
| weather(rain)    |         0.4180 |
+------------------+----------------+
| weather(sun)     |         0.0564 |
+------------------+----------------+
| weather(snow)    |         0.0157 |
+------------------+----------------+
| weather(fog)     |         0.0228 |
+------------------+----------------+
```

Insights:
Exact inference provided precise calculations revealing the direct effects of variables on each other, confirming that specific conditions like med precipitation and low wind significantly increase the likelihood of rainning weather levels.

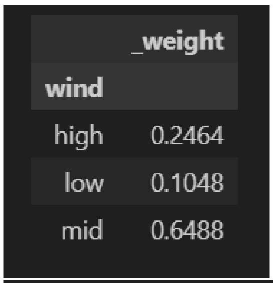
1.3 Approximate Inference Techniques

Methodology:
Different approximate inference techniques were applied for each question in Task 1.2 to test the robustness and efficiency of these methods in scenarios where exact computation might be impractical:

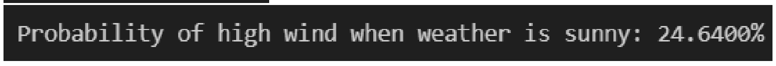
- **Likelihood weighting** for 1.2.1

  This technique improves upon rejection sampling by fixing observed nodes and sampling only the unobserved nodes, weighting each sample by the probability of the observed evidence. While it handles evidence more efficiently, it can suffer from significant weight variability, reducing its effectiveness as the number of evidence nodes increases.

  a)

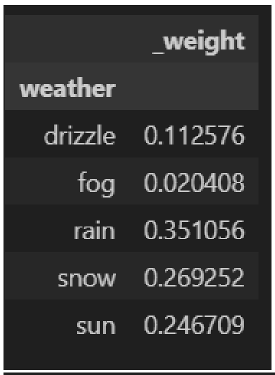


	_weight
wind	
high	0.2464
low	0.1048
mid	0.6488

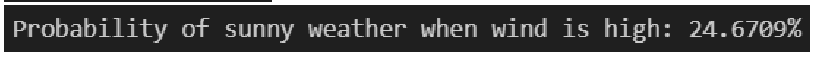


```
Probability of high wind when weather is sunny: 24.6400%
```

  b)



	_weight
weather	
drizzle	0.112576
fog	0.020408
rain	0.351056
snow	0.269252
sun	0.246709

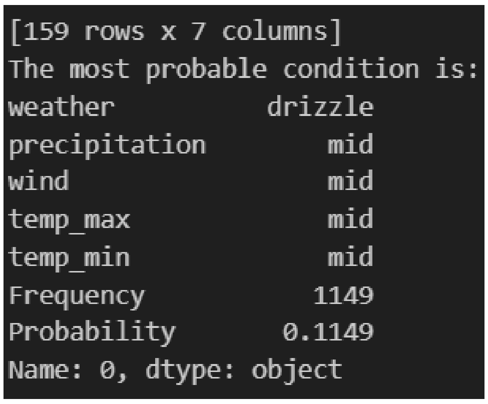


```
Probability of sunny weather when wind is high: 24.6709%
```

- **Rejection sampling** for 1.2.2

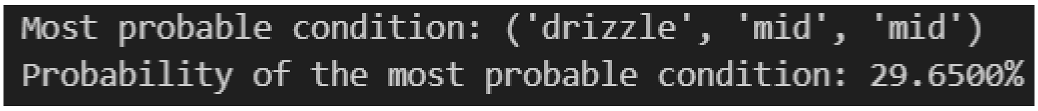
  This basic Markov Monte Carlo method generates potential samples from a broader distribution and rejects those that don't fit the target distribution. It's straightforward but often inefficient, as many samples may need to be discarded.

  a)



```
[159 rows x 7 columns]
The most probable condition is:
weather          drizzle
precipitation        mid
wind                 mid
temp_max             mid
temp_min             mid
Frequency           1149
Probability       0.1149
Name: 0, dtype: object
```

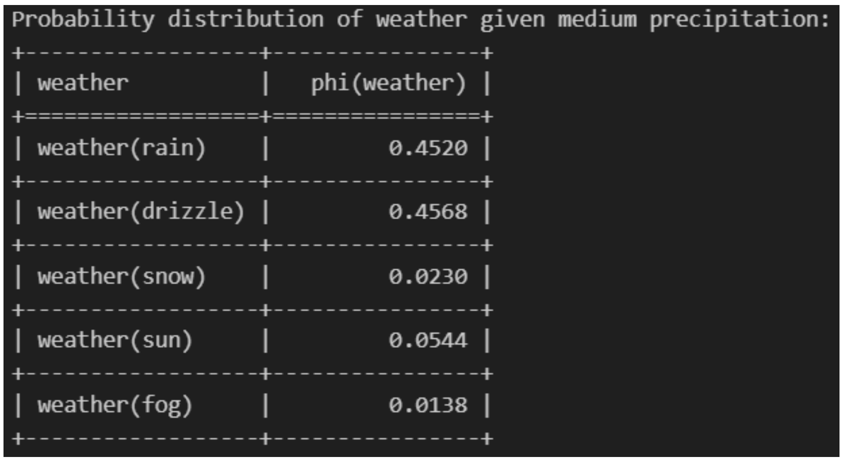
  b)



```
Most probable condition: ('drizzle', 'mid', 'mid')
Probability of the most probable condition: 29.6500%
```

- **Approx Inference** for 1.2.3

  It refers to methods for conducting approximate inference in Bayesian networks, where exact calculations are too complex or computationally demanding. This technique uses sampling, simulation, and other numerical strategies to estimate probability distributions quickly, trading some accuracy for increased efficiency. It's especially useful in large or intricate networks where rapid responses are more critical than perfect precision.

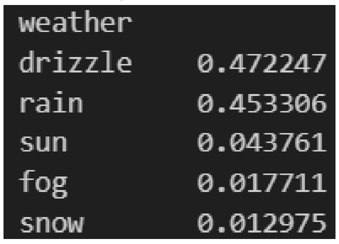


```
Probability distribution of weather given medium precipitation:
+------------------+----------------+
| weather          |   phi(weather) |
+==================+================+
| weather(rain)    |         0.4520 |
+------------------+----------------+
| weather(drizzle) |         0.4568 |
+------------------+----------------+
| weather(snow)    |         0.0230 |
+------------------+----------------+
| weather(sun)     |         0.0544 |
+------------------+----------------+
| weather(fog)     |         0.0138 |
+------------------+----------------+
```

- **Normal sampling** for 1.2.4

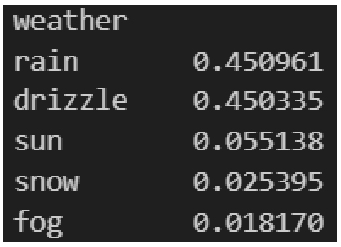
  Normal Sampling, often synonymous with forward sampling, directly generates samples from a Bayesian network by following the network's topological order. This method efficiently produces samples from the joint distribution of all variables without considering any observed evidence. It's particularly effective for visualizing and understanding the overall behavior of the network under various conditions, providing a straightforward approach to generate data that reflects the probabilities defined in the Bayesian model.

  a) Normalized probabilities of weather conditions given medium precipitation and wind being low or medium:



```
weather
drizzle    0.472247
rain       0.453306
sun        0.043761
fog        0.017711
snow       0.012975
```

  b) Comparison with only medium precipitation (no specific wind condition):



```
weather
rain       0.450961
drizzle    0.450335
sun        0.055138
snow       0.025395
fog        0.018170
```

Comparison and Results:
The approximate methods generally aligned well with the results from exact inference, though some discrepancies were noted due to sampling variability and biases inherent in each technique.

1.4 Exploration of Different Hierarchies

1.4.1 Create a Bayesian Network on the same data with two different hierarchies given below (ref. Figure 2 & 3).

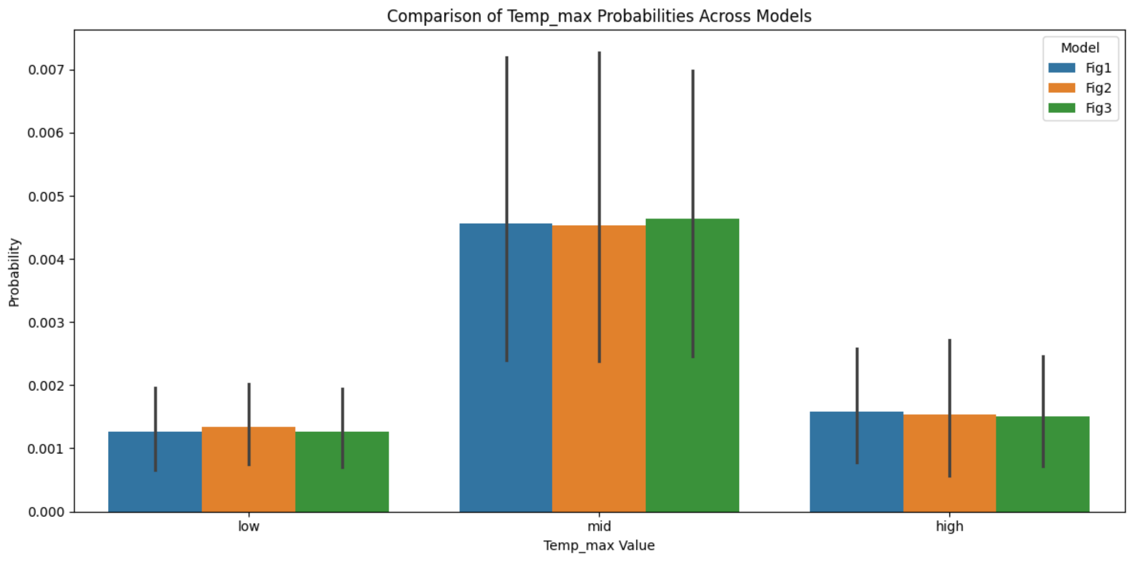
Methodology:
Two additional Bayesian Networks were constructed based on different hierarchical structures (Figure 2 & 3). Both exact and approximate inference methods were used to compare the joint probabilities across all three hierarchies.

Results Analysis:
Comparative analysis highlighted how the structural changes in the network affect predictions and probabilities, demonstrating that certain hierarchies might be more suited for specific types of predictive modeling based on the stability and accuracy of their outcomes.

See Assignment3.ipynb for further details.

1.4.2 Compare the joint probabilities from all the three-hierarchy using either exact or approximate inference? Discuss your results.

We made a plot titled "Comparison of Temp_max Probabilities Across Models" that visually represents the distribution of probabilities for the **temp_max** variable across three different Bayesian network models—Fig1, Fig2, and Fig3.



This plot directly addresses the task of comparing joint probabilities of the **temp_max** variable across different hierarchical Bayesian network models.

Analysis of Results:

1. **Consistency Across Models**: The graph indicates that the probability distributions for **temp_max** values ('low', 'mid', 'high') remain relatively consistent across the three models. This suggests that the different network structures do not significantly alter the probability distributions for **temp_max**, implying robustness across varying hierarchical arrangements.

2. **Error Bars and Variability**: The presence of error bars across all models shows the variability or the degree of uncertainty in the probability estimates. The similarity in the magnitude of these error bars across the models suggests that the level of uncertainty is comparable across different network configurations.

3. **Implications for Model Selection**: The similarity in probability distributions suggests that the choice of network hierarchy may not substantially impact the overall behavior of the **temp_max** variable. However, it may influence other aspects or specific details within the network, which might require additional analysis or a more detailed examination to uncover.

Task 2: Hierarchical Feature Analysis in Bayesian Networks

2.1. Bayesian Network Structure Learning

Providing insight on what criteria we are using to determine the feature relationships and hierarchy between variables:

To address the question of constructing a Bayesian Network model using the SPRICE dataset for maritime decision support, we should start by analyzing the correlation matrix provided and leveraging statistical methods to determine relationships and hierarchies among the variables.

Analysis of Correlation Data

The correlation matrix (Figure 5) highlights several key relationships:

- **Air Temperature and Relative Humidity** show some degree of negative correlation, suggesting that as temperature increases, humidity might decrease, and vice versa.
- **Wind Speed** and **Wind Direction** show little to no correlation, indicating that these variables might act independently in their influence on other variables.
- **Precipitation Intensity** has a strong correlation with **Precipitation Type**, which is intuitive as different types of precipitation (like rain or snow) will have different typical intensities.

Criteria for Bayesian Network Structure Learning

1. **Statistical Dependence**: Initial focus should be on variables that exhibit strong correlations, either positive or negative, as they likely influence each other. For example, the relationships between air temperature, relative humidity, and precipitation variables should be explored further.
2. **Causality Consideration**: While correlation does not imply causation, the understanding of the domain can help hypothesize potential causal relationships, which are crucial for Bayesian Networks. For instance, air temperature might causally affect precipitation intensity through the capacity of air to hold moisture.
3. **Data Coverage and Relevance**: Ensure that the features chosen for the network are relevant to the predictive goals and are adequately recorded in the dataset to avoid bias from missing or incomplete data.

Given a derived feature hierarchy, how can you employ Bayesian Network structure learning algorithms to construct a Bayes Net model? Explain the specific algorithm(s) you would select and justify your choice.

Bayesian Network Structure Learning Algorithms

To automate the structure learning from data, consider using algorithms such as:

- **Constraint-based algorithms** (e.g., PC algorithm): These algorithms use statistical tests to check for independence between variables and to form edges based on conditional dependencies.

- **Score-based algorithms** (e.g., Hill-Climbing, Tabu Search): These methods assign a score to different network structures based on how well they fit the data and search for the network that optimizes this score.

For this dataset, given its complexity and the nonlinear relationships likely involved with meteorological data:

- **Start with a Constraint-based approach** to discern initial structure ideas from statistical dependencies.
- **Refine with Score-based methods**, allowing for adjustments based on overall fit to the data, which might capture subtler interactions not immediately apparent from simple correlation analysis.

Algorithm Selection and Justification

- **PC Algorithm**: We can use this for an initial pass to establish a direct dependency graph based on conditional independence tests. This method is robust and can efficiently handle the conditional dependencies typical in environmental data.
- **Hill-Climbing Algorithm**: Following PC algorithm results, We can employ Hill-Climbing to iteratively refine the model by adjusting the structure to better fit the data, providing a balance between model complexity and data fitting.

Conclusion

The chosen methodology for Bayesian Network structure learning involves using both constraint and score-based algorithms in tandem to utilize the strengths of each. This hybrid approach helps to accurately model the intricate dependencies within maritime weather data, providing a powerful tool for prediction and decision-making in the maritime sector. By incorporating domain knowledge and statistical evidence, the Bayesian Network constructed will be robust, with significant predictive capabilities essential for operational and strategic decision-making.

2.2. Describing the techniques we would use to learn the conditional probability distribution (CPDs) for each possible combination of the features relationship for our Bayesian network:

To effectively learn the Conditional Probability Distributions (CPDs) for each possible combination of feature relationships in a Bayesian Network, a systematic approach incorporating both data-driven techniques and domain expertise is essential.

Techniques for Learning CPDs in Bayesian Networks:

1. **Parameter Learning with Complete Data**:
    - **Maximum Likelihood Estimation (MLE)**: If the dataset is complete (no missing values), MLE is a straightforward approach for estimating the probabilities that constitute the CPDs. MLE calculates the frequency of each variable configuration in the data and uses these frequencies as estimates of the probabilities.
    - **Usage**: This method is ideal when you have a large dataset with comprehensive coverage of all feature combinations, ensuring that the frequency estimates are reliable.
2. **Parameter Learning with Incomplete Data**:
    - **Expectation-Maximization (EM) Algorithm**: For datasets with missing values, the EM algorithm can be used to find estimates of the missing data and iteratively

refine the CPD estimates. EM alternates between inferring the missing values given the current parameters (E-step) and updating the parameters based on the inferred complete data (M-step).

- **Usage**: EM is particularly useful in complex real-world datasets where missing data is common, such as in meteorological records where some sensor data might be intermittently unavailable.

3. **Bayesian Estimation**:

    - **Bayesian Parameter Estimation**: Unlike MLE that provides point estimates, Bayesian estimation uses prior distributions (reflecting previous knowledge or assumptions about the parameters) and updates these with the observed data to produce posterior distributions of the CPD parameters.

    - **Usage**: This method is advantageous when prior domain knowledge is available or when the data is sparse. It helps in regularizing the CPD estimates, making them more robust to overfitting in cases of limited data.

4. **Utilizing Domain Knowledge**:

    - **Incorporating Expert Knowledge**: In cases where data is extremely sparse or certain feature combinations are rare, domain experts can provide insights that guide the setting of CPD parameters. This can be particularly relevant in specialized fields like maritime weather modeling.

    - **Usage**: Expert input can be crucial for defining plausible ranges or expected behaviors of certain variables under specific conditions, thereby refining the learned CPDs.

Implementation Strategy:

- **Data Preprocessing**: Ensure the data is clean and well-prepared, handling any missing or outlier values appropriately before applying learning algorithms.

- **Initial Parameter Setting**: Start with MLE or Bayesian Estimation to establish a baseline set of CPDs.

- **Iterative Refinement**: Apply the EM algorithm if the dataset has missing values, or iteratively adjust the CPDs based on new data and expert feedback.

- **Validation**: Continuously validate the learned CPDs against held-out data or through cross-validation to ensure their accuracy and robustness. Use performance metrics suited to the problem domain to assess the quality of the CPDs.

Conclusion:

Learning CPDs in a Bayesian Network for a dataset like the one provided for maritime decision support involves a blend of statistical techniques and expert insights. Each chosen method should align with the data characteristics and the specific requirements of the network's application, ensuring that the resulting CPDs are both accurate and practical for decision-making purposes. This approach not only enhances the predictive power of the Bayesian Network but also ensures that it remains adaptable and reliable under varying conditions.

2.3. Bayesian Inference and Analysis

With a constructed Bayesian network and learned parameters, perform similar inference Task 1.1, 1.2, and 1.3 for the new data with your choice of variables based on your constructed network you are analyzing.

Everything well explained and done in the Task2Assignment3.ipynb