ML LAB-1

(ID3 Algorithm)

NAME: NIKHIL GARUDA

SRN: PES2UG23CS195

SECTION: C

Comparative Analysis Report

a) Algorithm Performance

a. Which dataset achieved the highest accuracy and why?

The **Mushrooms** dataset achieved the highest accuracy with a perfect score of 1.00 (or 100%). This is likely because the dataset's features are highly discriminative. The distinct characteristics of poisonous and edible mushrooms, as represented by the features in the dataset, allow a classification model to create a clear separation between the two classes with no misclassifications. The confusion matrix for this dataset shows zero false positives and zero false negatives, indicating a flawless performance on the test set.

b. How does dataset size affect performance?

Generally, a larger dataset provides more examples for a model to learn from, which can lead to better performance. However, this is not always the sole determining factor. The **Nursery** dataset is the largest with 3,888 instances, but it has the lowest accuracy (0.98). The **Mushrooms** dataset has 2,438 instances and achieved perfect accuracy. This suggests that the *quality* and *discriminatory power* of the features are more critical than the sheer number of instances. The **Tic-Tac-Toe** dataset, with only 288 instances, shows a respectable accuracy of 0.86, indicating that even smaller datasets can perform well if the features are relevant and the problem is well-defined.

c. What role does the number of features play?

The number of features can significantly influence an algorithm's performance. The **Mushrooms** dataset has 22 features, the **Nursery** dataset has 8, and the **Tic-Tac-Toe** dataset has 9. While having more features can provide more information, it can also lead to overfitting if not all features are relevant. The perfect accuracy of the Mushrooms dataset, with its high number of features, suggests that all 22 features are highly informative and contribute to accurate classification. In contrast, the Nursery dataset's lower accuracy, despite having a large size, could indicate that its 8 features are not as discriminative or that the classes are more complex to separate.

b) Data Characteristics Impact

a. How does class imbalance affect tree construction?

Class imbalance can significantly affect the performance of a decision tree. When one class is much more frequent than others, a decision tree might become biased towards the majority class, leading to a high accuracy for that class but poor performance for the minority classes. This is evident in the Nursery dataset, where the "recommend" class has only 2 instances and the model achieved a precision, recall, and f1-score of 0.00. This suggests the model failed to correctly identify any of the "recommend" instances, likely due to the extreme class imbalance.

b. Which types of features (binary vs multi-valued) work better?

Based on the provided reports, binary features appear to be highly effective. The **Mushrooms** dataset, with its features representing attributes like 'edible' or 'poisonous' (a binary outcome), and the **Tic-Tac-Toe** dataset, with 'positive' or 'negative' outcomes, both performed well. Binary classification problems with clearly separable features can be highly effective. The **Nursery** dataset, which has multiple classes and potentially more complex, multi-valued features (e.g., 'very_crit' or 'foster'), resulted in lower performance, especially in minority classes. This suggests that while multi-valued features can capture more complex relationships, they can also make classification more challenging if the classes are not well-defined or if there is a class imbalance.

c) Practical Applications

a. For which real-world scenarios is each dataset type most relevant?

- **Mushrooms Dataset:** This type of data is most relevant for **medical or biological classification**, where a clear set of features can be used to distinguish between two outcomes (e.g., benign vs. malignant tumors, a virus vs. a bacterium). The high accuracy suggests it is suitable for applications where a definitive and highly accurate classification is critical.
- **Nursery Dataset:** This dataset is applicable to **multi-class classification problems in social sciences or policy-making**, such as school admissions, loan applications, or risk assessment. The presence of multiple, imbalanced classes reflects the complexity and nuance of real-world decision-making processes.
- **Tic-Tac-Toe Dataset:** This is relevant for **game theory and strategic decision-making analysis**. The dataset's structure could be used to model optimal moves or predict outcomes in games or other rule-based scenarios.

b. What are the interpretability advantages for each domain?

- **Mushrooms Dataset:** The high interpretability comes from the clear and distinct features. For example, if a model identifies a mushroom as poisonous due to a specific feature like 'odor=almond', this rule is easy to understand and apply. This is an

advantage in domains where understanding *why* a decision was made is as important as the decision itself.

- **Nursery Dataset:** The interpretability advantage here lies in the ability to understand the complex interactions between different criteria (e.g., 'housing' and 'finance') that lead to a final decision. While the model's overall performance might be lower, the resulting tree can provide valuable insights into the decision-making process, highlighting which factors are most influential.
- **Tic-Tac-Toe Dataset:** Interpretability is a major advantage for this dataset. The decision tree acts like a set of rules that can be easily translated into a human-readable strategy. For a human player, understanding which moves lead to a 'positive' outcome is straightforward, making the model a powerful teaching tool.

```
                        -> Value: 1.0
                           -> Leaf Node: Class 3.0
                        -> Value: 2.0
                           -> Leaf Node: Class 1.0
                  -> Value: 1.0
                     Attribute: housing
                       -> Value: 0.0
                          Attribute: finance
                            -> Value: 0.0
                               -> Leaf Node: Class 1.0
                            -> Value: 1.0
                               -> Leaf Node: Class 3.0
                       -> Value: 1.0
                          -> Leaf Node: Class 3.0
                       -> Value: 2.0
                          Attribute: form
                            -> Value: 0.0
                               -> Leaf Node: Class 1.0
```

```
✅ Evaluation Completed!
Accuracy: 1.0000

Confusion Matrix:
[[1257    0]
 [   0 1181]]

Classification Report:
              precision    recall  f1-score   support

           e       1.00      1.00      1.00      1257
           p       1.00      1.00      1.00      1181

    accuracy                           1.00      2438
   macro avg       1.00      1.00      1.00      2438
weighted avg       1.00      1.00      1.00      2438

PS C:\Users\nikhi>
```

```
🌳 Decision tree construction completed!

🎄 DECISION TREE STRUCTURE
============================================================
Attribute: odor
  -> Value: 0.0
     -> Leaf Node: Class 0.0
  -> Value: 1.0
     -> Leaf Node: Class 1.0
  -> Value: 2.0
     -> Leaf Node: Class 1.0
  -> Value: 3.0
     -> Leaf Node: Class 0.0
  -> Value: 4.0
     -> Leaf Node: Class 1.0
  -> Value: 5.0
     Attribute: spore-print-color
       -> Value: 0.0
          -> Leaf Node: Class 0.0
       -> Value: 1.0
          -> Leaf Node: Class 0.0
       -> Value: 2.0
          -> Leaf Node: Class 0.0
       -> Value: 3.0
          -> Leaf Node: Class 0.0
       -> Value: 4.0
          -> Leaf Node: Class 0.0
       -> Value: 5.0
          -> Leaf Node: Class 1.0
       -> Value: 7.0
          Attribute: habitat
            -> Value: 0.0
               Attribute: gill-size
                 -> Value: 0.0
                    -> Leaf Node: Class 0.0
                 -> Value: 1.0
```

✅ Evaluation Completed!
Accuracy: 0.8611

Confusion Matrix:
[[78 17]
 [23 170]]

Classification Report:

	precision	recall	f1-score	support
negative	0.77	0.82	0.80	95
positive	0.91	0.88	0.89	193
accuracy			0.86	288
macro avg	0.84	0.85	0.85	288
weighted avg	0.86	0.86	0.86	288

```
PS C:\Users\nikhi> & C:/Users/nikhi/AppData/Local/Programs/Python/Python313/python.exe c:/Users/nikhi/ml1.py
Starting decision tree construction...
🌳 Decision tree construction completed!

🌲 DECISION TREE STRUCTURE
============================================================
Attribute: middle-middle-square
  -> Value: 0.0
     Attribute: bottom-left-square
       -> Value: 0.0
          Attribute: top-right-square
            -> Value: 1.0
               -> Leaf Node: Class 0.0
            -> Value: 2.0
               -> Leaf Node: Class 1.0
       -> Value: 1.0
          Attribute: top-right-square
            -> Value: 0.0
               -> Leaf Node: Class 0.0
            -> Value: 1.0
               -> Leaf Node: Class 0.0
            -> Value: 2.0
               Attribute: bottom-right-square
                 -> Value: 0.0
                    Attribute: top-left-square
                      -> Value: 1.0
                         -> Leaf Node: Class 0.0
                      -> Value: 2.0
                         -> Leaf Node: Class 1.0
                 -> Value: 1.0
                    Attribute: middle-left-square
                      -> Value: 0.0
                         -> Leaf Node: Class 0.0
                      -> Value: 1.0
                         -> Leaf Node: Class 1.0
                      -> Value: 2.0
                         -> Leaf Node: Class 0.0
                 -> Value: 2.0
                    Attribute: middle-right-square
                      -> Value: 0.0
                         Attribute: top-left-square
                           -> Value: 1.0
```

```
    not_recom       1.00      1.00      1.00      1320
     priority       0.95      0.98      0.96      1272
    recommend       0.00      0.00      0.00         2
   spec_prior       0.99      0.96      0.98      1190
   very_recom       0.81      0.81      0.81       104

     accuracy                           0.98      3888
    macro avg       0.75      0.75      0.75      3888
 weighted avg       0.98      0.98      0.98      3888
```