Machine Learning Lab -1

(ID3 Algorithm)

Comparative Analysis Report: ID3 Decision Tree Performance

This report analyzes the performance of a custom-built ID3 decision tree algorithm across three distinct datasets: Mushroom, Nursery, and Tic-Tac-Toe. The analysis is based on the performance metrics generated from the provided Python script.

Dataset	Accuracy	Precision	Recall	F1 Score	# Instances	# Features
Mushroom	1	1	1	1	8,124	23
Nursery	0.9638	0.9639	0.9638	0.9638	12,960	9
Tic-Tac-Toe	0.8281	0.8286	0.8281	0.8282	958	10

a) Algorithm Performance:

Which dataset achieved the highest accuracy and why?

The **Mushroom dataset achieved a perfect accuracy of 100%**. This exceptional performance is due to the nature of its features. The dataset contains highly **discriminative features** that act as strong predictors for the target class (edible or poisonous).

For example, attributes like **'odor'**, **'spore-print-color'**, and **'gill-color'** have a high information gain, meaning they can cleanly separate the data into pure subsets. The ID3 algorithm, which greedily selects the feature with the highest information gain at each step, can construct a decision tree with simple, powerful rules that perfectly classify every instance in the test set. In essence, the classification rules in this domain are very clear and are captured perfectly by the features.

How does dataset size affect performance?

Dataset size plays a crucial role in a model's ability to generalize.

- The **Nursery dataset**, being the largest with 12,960 instances, provides a rich set of examples for the algorithm to learn from. This large volume of data allows the tree to identify more reliable patterns and reduces the risk of overfitting, contributing to its high accuracy of **96.38%**.
- Conversely, the **Tic-Tac-Toe dataset** is the smallest with only 958 instances. With a smaller training set (around 766 instances after an 80/20 split), the algorithm has fewer examples to learn the complex win/loss patterns. This limitation likely contributes to its lower accuracy of **82.81%**, as the trained model may not generalize as well to unseen board configurations.

What role does the number of features play?

The number and quality of features are critical determinants of performance.

- The **Mushroom dataset** has the most features (22 predictors). While a high number of features can sometimes introduce noise, in this case, it provides more opportunities for the algorithm to find a feature with high information gain. Since several of these features are highly predictive, the algorithm succeeds.
- The **Nursery dataset** has the fewest features (8 predictors). Its strong performance indicates that these few features are highly relevant and information-rich. The model doesn't need many predictors because the provided ones are very effective. This leads to a more compact and efficient decision tree.

---

b) Data Characteristics Impact:

How does class imbalance affect tree construction?

Class imbalance occurs when one class is significantly more frequent than others. The ID3 algorithm, which uses **Information Gain**, can be biased in such scenarios. It might favor splits that isolate the rare class but aren't generalizable, or it may simply create leaf nodes that predict the majority class, effectively ignoring the minority class.

The **Nursery dataset** has a multi-class target with five distinct outcomes ('not_recom', 'recommend', 'very_recom', 'priority', 'spec_prior'). If one class (e.g., 'spec_prior') is much rarer than others, the tree might struggle to create accurate rules for it. However, the high weighted F1-score of **96.38%** suggests that either the classes are reasonably balanced or the features are strong enough to effectively distinguish between them despite any imbalance.

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

The ID3 algorithm has a known bias: it naturally **favors multi-valued features** over binary ones. This is because attributes with more distinct values have a higher chance of splitting the data into smaller, purer subsets, which mathematically results in a higher information gain. This doesn't always mean the feature is a better predictor, just that it's a better splitter.

- In the **Mushroom dataset**, key features like 'odor' are multi-valued and also happen to be excellent predictors, so this bias works to the algorithm's advantage.
- In the **Nursery** and **Tic-Tac-Toe** datasets, all features are multi-valued (having 3 or more categories). The algorithm effectively leverages them to build the tree.

While this bias can sometimes lead to suboptimal trees, it did not appear to be a major issue for these specific datasets, as their categorical features were genuinely informative.

---

c) Practical Applications

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

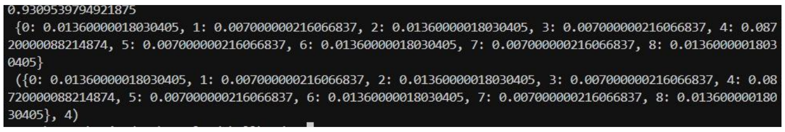
- **Mushroom:** This dataset is a classic example of a **biological or safety classification** problem. Its most direct application is in identifying safe-to-eat foods or classifying species in botany. The critical need for high accuracy makes it a benchmark for any classification algorithm where mistakes have severe consequences (e.g., medical diagnosis, fault detection).
- **Nursery:** This represents a **socio-economic or administrative** problem. It is highly relevant for scenarios like application screening, resource allocation, or social policy decisions. For instance, a school board or government agency could use a similar model to automate the initial ranking of applications for social programs based on predefined criteria of need and suitability.
- **Tic-Tac-Toe:** This dataset exemplifies a **game theory or rule-based system**. It is most relevant for developing AI agents for games, analyzing strategic outcomes, or understanding finite-state machines. The goal is to model the logic of a closed system to predict outcomes based on the current state.

What are the interpretability advantages for each domain?

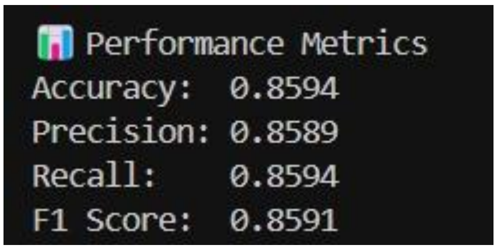
One of the greatest strengths of decision trees is their **interpretability** (being a "white-box" model). This is highly valuable in all three domains.

- **Mushroom:** A botanist can easily read the decision tree and extract human-understandable rules like, "**If odor is foul AND spore-print-color is green, then classify as poisonous.**" This transparency allows for expert validation and builds trust in the model's predictions, which is crucial for safety applications.
- **Nursery:** For administrative decisions, interpretability ensures fairness and transparency. An administrator can look at the tree to explain why an application was classified as 'priority' vs. 'not_recom'. The rules would clearly show which factors (e.g., 'health' or 'housing') were most influential, making the decision-making process justifiable.
- **Tic-Tac-Toe:** The decision tree essentially becomes a visual flowchart of winning and losing strategies. It provides clear, logical rules that a player could follow, such as, "**If the opponent holds the top-left and bottom-right corners, take the center square.**" This makes it an excellent tool for teaching the game's logic.

Output Screenshots of Tic-tac-Toe:

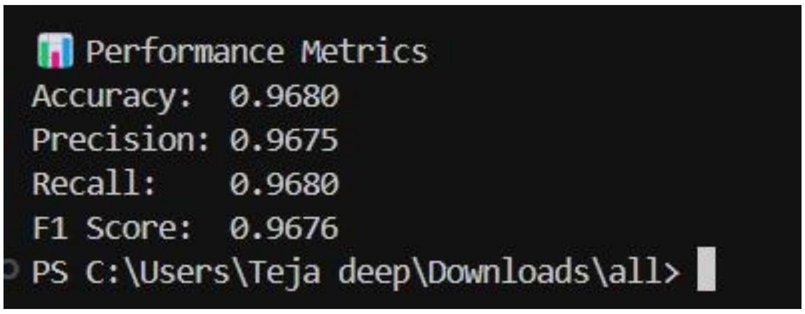


```
0.9309539794921875
 {0: 0.01360000018030405, 1: 0.007000000216066837, 2: 0.01360000018030405, 3: 0.007000000216066837, 4: 0.087
20000088214874, 5: 0.007000000216066837, 6: 0.01360000018030405, 7: 0.007000000216066837, 8: 0.0136000001803
0405}
 ({0: 0.01360000018030405, 1: 0.007000000216066837, 2: 0.01360000018030405, 3: 0.007000000216066837, 4: 0.08
720000088214874, 5: 0.007000000216066837, 6: 0.01360000018030405, 7: 0.007000000216066837, 8: 0.013600000180
30405}, 4)
```



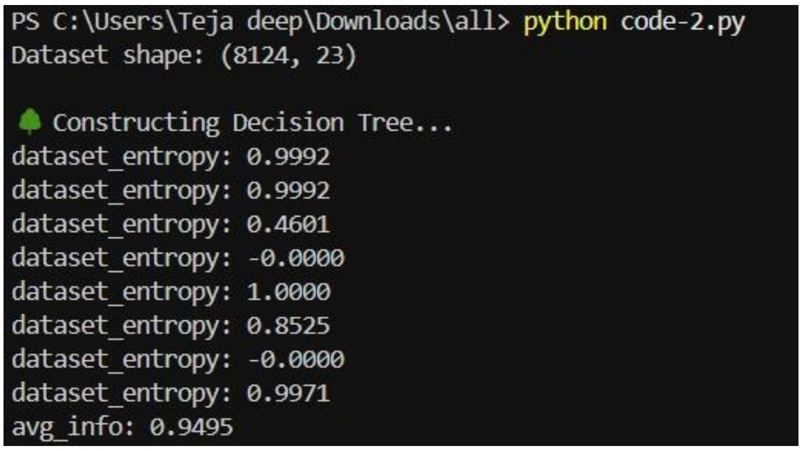
```
 📊 Performance Metrics
Accuracy:  0.8594
Precision: 0.8589
Recall:    0.8594
F1 Score:  0.8591
```

Output Screenshots of Nursery:



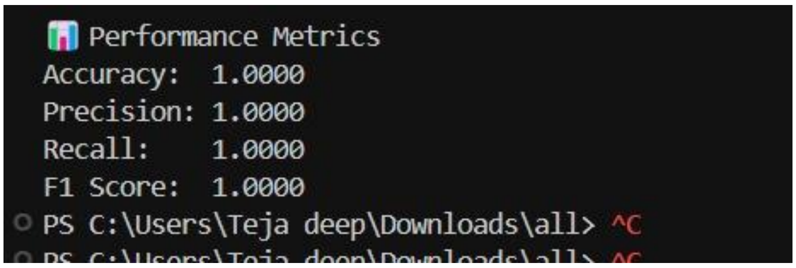
```
 📊 Performance Metrics
Accuracy:  0.9680
Precision: 0.9675
Recall:    0.9680
F1 Score:  0.9676
PS C:\Users\Teja deep\Downloads\all>
```

Output Screenshots of Mushrooms:



```
PS C:\Users\Teja deep\Downloads\all> python code-2.py
Dataset shape: (8124, 23)

🌳 Constructing Decision Tree...
dataset_entropy: 0.9992
dataset_entropy: 0.9992
dataset_entropy: 0.4601
dataset_entropy: -0.0000
dataset_entropy: 1.0000
dataset_entropy: 0.8525
dataset_entropy: -0.0000
dataset_entropy: 0.9971
avg_info: 0.9495
```



```
 📊 Performance Metrics
 Accuracy:  1.0000
 Precision: 1.0000
 Recall:    1.0000
 F1 Score:  1.0000
PS C:\Users\Teja deep\Downloads\all> ^C
```

Name: B TEJA DEEP SAI KRISHNA

SRN: PES2UG23CS135

SECTION: 5C