

Name: Mohithkumar M S	SRN: PES2UG23CS917
Class : 5c	ML LAB

#### a) Algorithm Performance

a. mushrooms.csv has achieved the highest accuracy.

- Odor produces high Information gain
- Trees can classify “poisonous vs edible”
- Entropy of the full dataset is 1. After splitting on odor, subsets are nearly pure (entropy=0) so Information gain is maximized
- We can reach to conclusion by only a few levels of splitting

b. How does dataset size affect performance?

More is the dataset size more is the generalization which thus improves the performance of the tree

c. What role does the number of features play?

More features → more split choices.

Mushrooms: few features give very high IG.

TicTacToe: all features needed, each with small IG.

Nursery: multi-valued features → deeper trees needed.

#### B) Data Characteristics Impact

- Class Imbalance effect:  
It lowers the entropy since one class dominates, so information gain favours the splits that separate the majority. As a result the tree is biased towards majority class and recall for minority class drops
- Binary and multi-valued features  
Binary – Information gain per split is smaller, need more records (TicTacToe)  
Multi-valued – can yield higher Information gain if one value is strongly predictive (mushrooms)

#### C) Practical Applications:

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

TicTacToe-game state

Mushrooms- purity check in terms of health

Nursery-priority ranking

- What are the interpretability advantages for each domain?

TicTacToe → rule-based game strategies.

Mushrooms → simple, safety rules (“odor=foul → poisonous”).

Nursery → transparent multi-criteria prioritization paths

- How would you improve performance for each dataset

**TicTacToe:** deeper trees, engineered features (two-in-a-row).

**Mushrooms:** prune redundant features, ensembles for probability calibration.

**Nursery:** handle imbalance (weights/SMOTE), use ensembles, combine feature groups

## 1)TicTacToe.csv

```
PS C:\Users\HP\Desktop\SEM 5\ML LAB\Lab 2\EC_SC_PES2UG23CS917_Lab3> python test.py --ID CAMPUS_SECTION_SSM_Lab3 --data tictactoe.csv
Running tests with PYTORCH framework
=====
target column: 'Class' (last column)
Original dataset info:
Shape: (958, 10)
Columns: ['top-left-square', 'top-middle-square', 'top-right-square', 'middle-left-square', 'middle-middle-square', 'middle-right-square', 'bottom-left-square', 'bottom-middle-square', 'bottom-right-square', 'Class']

First few rows:

top-left-square: ['x' 'o' 'b'] -> [2 1 0]

top-middle-square: ['x' 'o' 'b'] -> [2 1 0]

top-right-square: ['x' 'o' 'b'] -> [2 1 0]

Class: ['positive' 'negative'] -> [1 0]

Processed dataset shape: torch.Size([958, 10])
Number of features: 9
Features: ['top-left-square', 'top-middle-square', 'top-right-square', 'middle-left-square', 'middle-middle-square', 'middle-right-square', 'bottom-left-square', 'bottom-middle-square', 'bottom-right-square']
Target: Class
construction completed using PYTORCH!
```

### OVERALL PERFORMANCE METRICS

```
=====
Accuracy:          0.8730 (87.30%)
Precision (weighted): 0.8741
Recall (weighted):  0.8730
F1-Score (weighted): 0.8734
Precision (macro):  0.8590
Recall (macro):     0.8638
F1-Score (macro):   0.8613
```

### TREE COMPLEXITY METRICS

```
=====
Maximum Depth:      7
Total Nodes:         281
Leaf Nodes:          180
Internal Nodes:      101
```

## 2)mushrooms.csv

```
PS C:\Users\HP\Desktop\SEM 5\ML LAB\Lab 2\EC_5C_PES2UG23CS917_Lab3> python test.py --ID CAMPUS_SECTION_SRN_Lab3 --data mushrooms.csv
Running tests with PYTORCH framework
=====
target column: 'class' (last column)
Original dataset info:
Shape: (8124, 23)
Columns: ['cap-shape', 'cap-surface', 'cap-color', 'bruises', 'odor', 'gill-attachment', 'gill-spacing', 'gill-size', 'gill-color', 'stalk-shape', 'stalk-root', 'stalk-surface-above-ring', 'stalk-surface-below-ring', 'stalk-color-above-ring', 'stalk-color-below-ring', 'veil-type', 'veil-color', 'ring-number', 'ring-type', 'spore-print-color', 'population', 'habitat', 'class']

First few rows:

cap-shape: ['x' 'b' 's' 'f' 'k'] -> [5 0 4 2 3]
cap-surface: ['s' 'y' 'f' 'g'] -> [2 3 0 1]


cap-color: ['n' 'y' 'w' 'g' 'e'] -> [4 9 8 3 2]

class: ['p' 'e'] -> [1 0]

Processed dataset shape: torch.Size([8124, 23])
Number of features: 22
Features: ['cap-shape', 'cap-surface', 'cap-color', 'bruises', 'odor', 'gill-attachment', 'gill-spacing', 'gill-size', 'gill-color', 'stalk-shape', 'stalk-root', 'stalk-surface-above-ring', 'stalk-surface-below-ring', 'stalk-color-above-ring', 'stalk-color-below-ring', 'veil-type', 'veil-color', 'ring-number', 'ring-type', 'spore-print-color', 'population', 'habitat']
Target: class
Framework: PYTORCH
Data type: <class 'torch.Tensor'>

=====
DECISION TREE CONSTRUCTION DEMO
=====
Total samples: 8124
Training samples: 6499
Testing samples: 1625
```

### Constructing decision tree using training data...

 Decision tree construction completed using PYTORCH!

#### OVERALL PERFORMANCE METRICS

```
=====
Accuracy:          1.0000 (100.00%)
Precision (weighted): 1.0000
Recall (weighted):  1.0000
F1-Score (weighted): 1.0000
Precision (macro):  1.0000
Recall (macro):     1.0000
F1-Score (macro):   1.0000
```

#### TREE COMPLEXITY METRICS

```
=====
Maximum Depth:      4
Total Nodes:         29
Leaf Nodes:          24
Internal Nodes:      5
```

## 3)Nursery.csv

```

PS C:\Users\HP\Desktop\SEM 5\ML LAB\Lab_2\EC_PES2UG23CS917_Lab3> python test.py --ID CAMPUS_SECTION_SRN_Lab3 --data nursery.csv
Running tests with PYTORCH framework
=====
target column: 'class' (last column)
Original dataset info:
Shape: (12960, 9)
Columns: ['parents', 'has_nurs', 'form', 'children', 'housing', 'finance', 'social', 'health', 'class']

First few rows:

parents: ['usual' 'pretentious' 'great_pret'] -> [2 1 0]

has_nurs: ['proper' 'less_proper' 'improper' 'critical' 'very_crit'] -> [3 2 1 0 4]

form: ['complete' 'completed' 'incomplete' 'foster'] -> [0 1 3 2]


class: ['recommend' 'priority' 'not_recom' 'very_recom' 'spec_prior'] -> [2 1 0 4 3]


Processed dataset shape: torch.Size([12960, 9])
Number of features: 8
Features: ['parents', 'has_nurs', 'form', 'children', 'housing', 'finance', 'social', 'health']
Target: class
Framework: PYTORCH
Data type: <class 'torch.Tensor'>

=====
DECISION TREE CONSTRUCTION DEMO
=====
Total samples: 12960
Training samples: 10368
Testing samples: 2592

```

Constructing decision tree using training data...


 Decision tree construction completed using PYTORCH!

 OVERALL PERFORMANCE METRICS

```

=====
Accuracy:                0.9867 (98.67%)
Precision (weighted): 0.9876
Recall (weighted):      0.9867
F1-Score (weighted):   0.9872
Precision (macro):      0.7604
Recall (macro):         0.7654
F1-Score (macro):      0.7628

```

 TREE COMPLEXITY METRICS

```

=====
Maximum Depth:          7
Total Nodes:            952
Leaf Nodes:             680
Internal Nodes:         272

```