

NAME: ROHAN SURESH

SRN: PES1UG23AM240

AIML 5th SEM D SECTION

ML LAB-03

Mushrooms Dataset:

```
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-c
olor', 'stalk-shape', 'stalk-root', 'stalk-surface-above-ring', 'stalk-surface-below-ring', 'stalk-color-above-ring', 'stalk-c
olor-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:          0.6898 (68.98%)
Precision (weighted): 0.7400
Recall (weighted):  0.6898
F1-Score (weighted): 0.6725
Precision (macro):   0.7402
Recall (macro):      0.6894
F1-Score (macro):    0.6724

● TREE COMPLEXITY METRICS
=====
Maximum Depth:      7
Total Nodes:         36
Leaf Nodes:          15
Internal Nodes:       21
```

TicTacToe Dataset:

```
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-r
ight-square', 'bottom-left-square', 'bottom-middle-square', 'bottom-right-square']
Target: class
Framework: PYTORCH
Data type: <class 'torch.Tensor'>

=====
DECISION TREE CONSTRUCTION DEMO
=====
Total samples: 958
Training samples: 766
Testing samples: 192

Constructing decision tree using training data...

● Decision tree construction completed using PYTORCH!

📊 OVERALL PERFORMANCE METRICS
=====
Accuracy:          0.6304 (63.04%)
Precision (weighted): 0.5945
Recall (weighted):   0.6304
F1-Score (weighted): 0.6055
Precision (macro):    0.5335
Recall (macro):       0.5255
F1-Score (macro):    0.5208

● TREE COMPLEXITY METRICS
=====
Maximum Depth:      7
Total Nodes:         961
Leaf Nodes:          527
Internal Nodes:       434
```

Nursery Dataset:

```
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: 0.1270 (12.70%)
Precision (weighted): 0.1518
Recall (weighted): 0.1270
F1-Score (weighted): 0.1315
Precision (macro): 0.0944
Recall (macro): 0.0780
F1-Score (macro): 0.0813

TREE COMPLEXITY METRICS
=====
Maximum Depth: 7
Total Nodes: 5487
Leaf Nodes: 4200
Internal Nodes: 1287
```

Report on Decision Tree Construction and Performance

a) Algorithm Performance

- **Highest Accuracy:** Mushroom dataset (**68.98%**) because features (odour, gill colour, etc.) are strongly correlated with class.
- **Dataset Size:** Larger size (Nursery, 12,960 samples) did not help; feature quality mattered more.
- **Features:** More features improve performance only when they are relevant. Mushroom (22 features) outperformed Nursery (8 features).

b) Data Characteristics

- **Class Imbalance:** Nursery suffers from imbalance (rare `very_recom` class), leading to poor accuracy. Balanced datasets (Mushroom, Tic-Tac-Toe) perform better.
- **Binary vs Multi-valued Features:** Binary/low-cardinality features (Mushroom, Tic-Tac-Toe) yield clearer splits than multi-valued ones (Nursery)

c) Practical Applications

- **Nursery:** Admission decisions, education planning.
- **Tic-Tac-Toe:** Simple game AI, teaching ML concepts.
- **Mushroom:** Food safety, biology.

Interpretability: Rules are transparent (e.g., “odour=foul \rightarrow poisonous”), useful for policy, game strategy, and safety-critical tasks.

d) Improvements

- **Nursery:** Balance classes, engineer features, use ensembles.
- **Tic-Tac-Toe:** Add sequence features, prune trees, ensembles.
- **Mushroom:** Feature selection, Random Forests for >95% accuracy.