Date: 19-08-2025

Machine Learning- Lab 3 Submission

Name: C Bhargav **SRN**: PES2UG23CS137 **Section**: C

1. Output of Dataset -1 (Mushroom.csv)

```
============================================================
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
```

Short Description:

Out of 8,124 total samples, the dataset was split into 6,499 training samples and 1,625 testing samples.

The constructed decision tree achieved perfect performance, with 100% accuracy, precision, recall, and F1-score on the test set. This indicates that the dataset's attributes provide sufficient information for the decision tree to completely separate edible from poisonous mushrooms.

Date: 19-08-2025

2. Output of Dataset – 2 (TicTacToe.csv)

```
============================================================
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:             0.8730 (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
```

Short Description:

The dataset contained 958 total samples, with a split of 766 training samples and 192 testing samples.

The Decision Tree classifier was trained and evaluated on the **Tic-Tac-Toe Endgame dataset**, which contains board states represented by 9 categorical features (each square being x, o, or b) with the class label indicating whether the configuration is a **win** (positive) or **not a win** (negative).

The decision tree achieved an accuracy of 87.3% on the test set, with balanced precision, recall, and F1-scores (around 0.86–0.87). These results show that the model is effective at learning winning patterns, though the imperfect performance indicates the dataset has overlapping board configurations that are more challenging to separate.

Date: 19-08-2025

3. Output of Dataset – 3 (Nursery.csv)

```
============================================================
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
```

Short Description:

The dataset contained **12,960 samples**, divided into **10,368 training samples** and **2,592 testing samples**.

The trained decision tree achieved an **accuracy of 98.67%**, with weighted precision, recall, and F1-scores all close to **0.987**, showing that the classifier performs exceptionally well on the dataset. However, the **macro-averaged scores** (Precision: 0.760, Recall: 0.765, F1: 0.763) indicate some imbalance in classification performance across the different target classes, suggesting that the model performs better on majority classes compared to minority ones.

---------- X ----------