Assignment II Report

By: Kaushik Nadimpalli

1. (Autograder Score, 20 points) Your code will be auto-graded and cross-checked with other submissions. The auto-grader will evaluate your code on several different data sets to perform a sanity check. In order to ensure that your code passes the auto-grader, ensure that you **do not modify the function headers**. In addition, do not hard code any values (such as y = 0 and 1) and make your code as general as possible.

* There are 2 files in the zip folder
  + My decision tree implementation
    - File: decision\_tree.py
  + Sci-Kit Learn Implementation
    - decision\_tree\_scikit.py
* The two files above are provided in the zip folder
  + Note that some parts of the code are commented out, but the main functions calculating entropy, mutual information, and our ID3 algorithm are not. As long as the datasets are loaded properly, those algorithms will work normally for sanity check (can confirm as we see in Part E with another dataset). The commented-out code is for printing the plots (part b) and confusion matrices for specific depths (part c) and testing new dataset (part e). If need be, you can remove the comments appropriately for whichever part needed to be tested.

**Part B**

Monk 1 Dataset - Decision Tree and Plot

A close up of text on a white background

Description automatically generated

A close up of a map

Description automatically generated

Monk 2 Dataset – Decision Tree and Plot

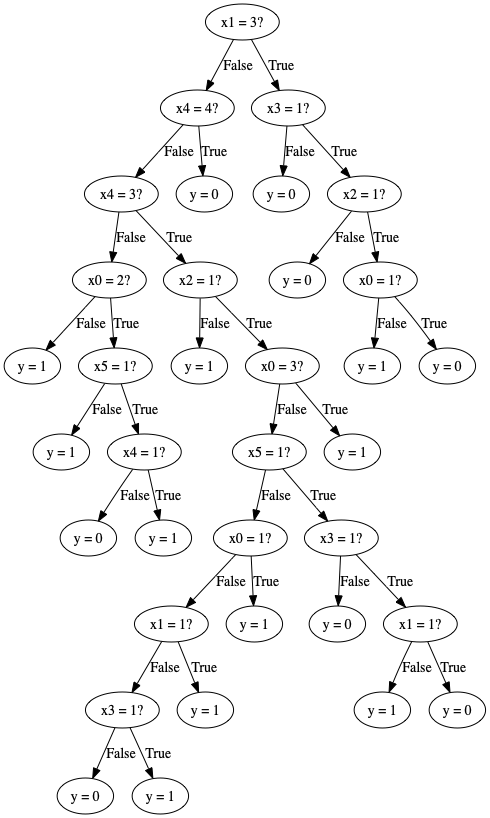
A close up of a map

Description automatically generated

A close up of a map

Description automatically generated

Monk 3 Dataset - Decision Tree and Plot



A close up of a map

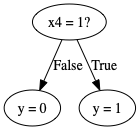
Description automatically generated

**Part C**

For Monks 1, below are the visualized trees and Confusion Matrices for fixed depths d = 1,3,5

Depth = 1

Tree



Confusion Matrix

[[216 0]

[108 108]]

Depth = 3

Tree

A close up of a logo

Description automatically generated

Confusion Matrix

[[144 72]

[0 216]]

Depth = 5

Tree

A close up of text on a white background

Description automatically generated

Confusion Matrix

[[156 60]

[12 204]]

**Part D**

For this part, we used SciKit-Learn’s DecisionTreeClassifier to build our decision tree for depths of 1, 3, and 5. Below are the visualizations for the trees as well as their confusion matrices for trees of those depths

Depth = 1

A close up of a sign

Description automatically generated

Confusion Matrix

[[216 0]

[108 108]]

Depth = 3

A close up of a piece of paper

Description automatically generated

Confusion Matrix

[[144 72]

[0 216]]

Depth = 5

A close up of a piece of paper

Description automatically generated

Confusion Matrix

[[168 48]

[24 192]]

**Part E**

For this part, we are working with another dataset. We will be testing our decision tree vs SciKit-Learn’s decision tree, and once again compare the confusion matrices yielded for both our tree and their tree for depths of 1, 3, and 5.

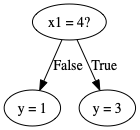
The dataset we chose to evaluate re-evaluate parts C and D is the Hayes-Roth dataset from UCI.

There are 5 attributes and 160 instances.

<https://archive.ics.uci.edu/ml/datasets/Hayes-Roth>

*Part C for new dataset*

Depth 1

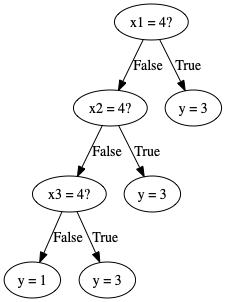
**

Confusion Matrix

[[14 0]

[13 0]]

Depth 3

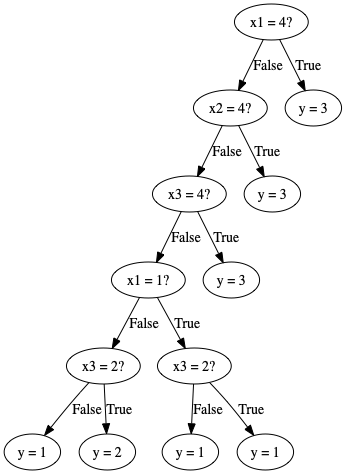
**

Confusion Matrix

[[14 0]

[10 0]]

Depth 5

**

Confusion Matrix

[[14 0]

[7 0]]

*Part D for new dataset*

Here, once again, let us take a look at SciKit-Learn’s decision tree’s confusion matrices below for the 3 depths on the Hayes Roth dataset.

Depth 1

[[14 0 0]

[13 0 0]

[0 0 1]]

Depth 3

[[14 0 0]

[13 0 0]

[0 0 1]]

Depth 5

[[10 4 0]

[5 8 0]

[0 0 1]]

**Final Analysis**

The purpose of this assignment is to implement our own decision tree using the ID3 algorithm. We are building another tree using scikit-learn’s decision tree classifier, and then compare the results yielded by both. The core difference that can be seen between our implemented algorithm and the one from scikit-learn is that while scikit-learn’s tree classifier prunes by finding a threshold (CART) and going from there. In our case, we are splitting our data into optimal value-pairs i.e. separate the good values from the bad ones. We are focused on finding the best value for the input variable.

We can also see the confusion matrices which showcase 4 different predictions vs actual comparisons. They are false-negative, false-positive, true-negative, and true-positive. With a depth of 1, we notice that the matrices are same for our algorithm and the classifier. However, as we prune and go further, we can see a noticeable difference between the confusion matrices values.

Essentially, we are comparing two types of implementations: ID3 and Cart. In our ID3 built tree, we are splitting based on Mutual Information, which is the reduction in entropy (which we also calculated) between parent node and children node (sum of children nodes that is. Contrarily, with Cart makes its splits based on the subsets that minimize Gini impurity, which measure of how often a randomly chosen element from the set would be incorrectly labeled if it was randomly labeled according to the distribution of labels in the subset.