Machine Learning Programming Assignment – 1

Name: Nikhilesh Prabhakar
NetID: NXP190053

Problem: Implement a **fixed-depth decision tree algorithm**, that is, the input to the `ID3` algorithm will include the training data and **maximum depth of the tree** to be learned. The code skeleton as well as data sets for this assignment can be found on e-Learning.

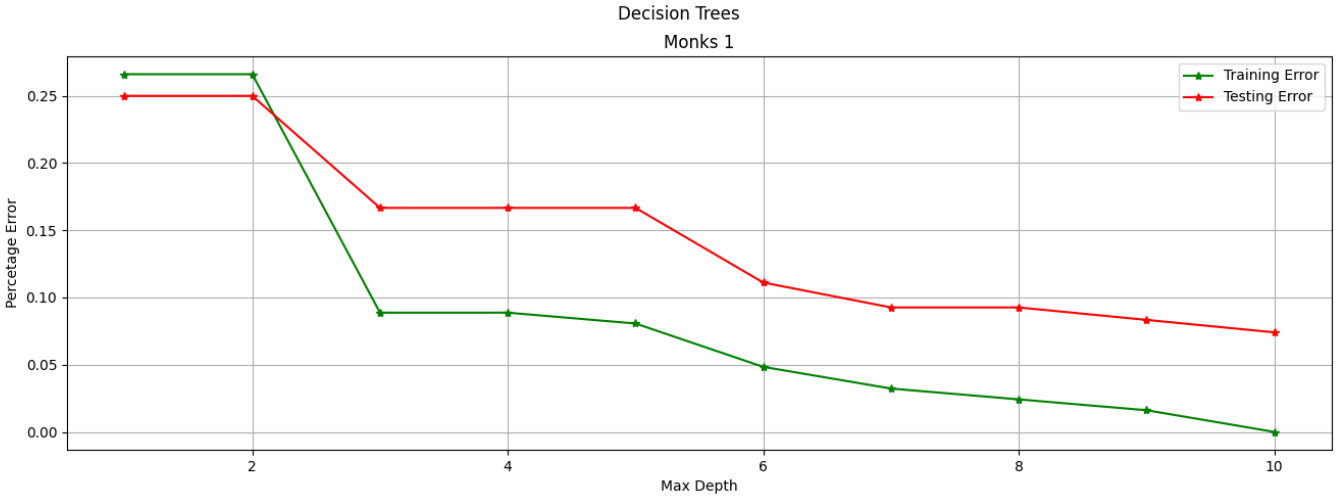
Data Sets: The data sets (in the folder `./data/`) are obtained from the UCI Repository and are collectively the MONK's Problem. These problems were the basis of a first international comparison of learning algorithms[1]. The training and test files for the three problems are named `monks-X.train` and `monks-X.test`. There are six attributes/features (columns 2–7 in the raw files), and the class labels (column 1). There are 2 classes. Refer to the file `./data/monks.names` for more details.

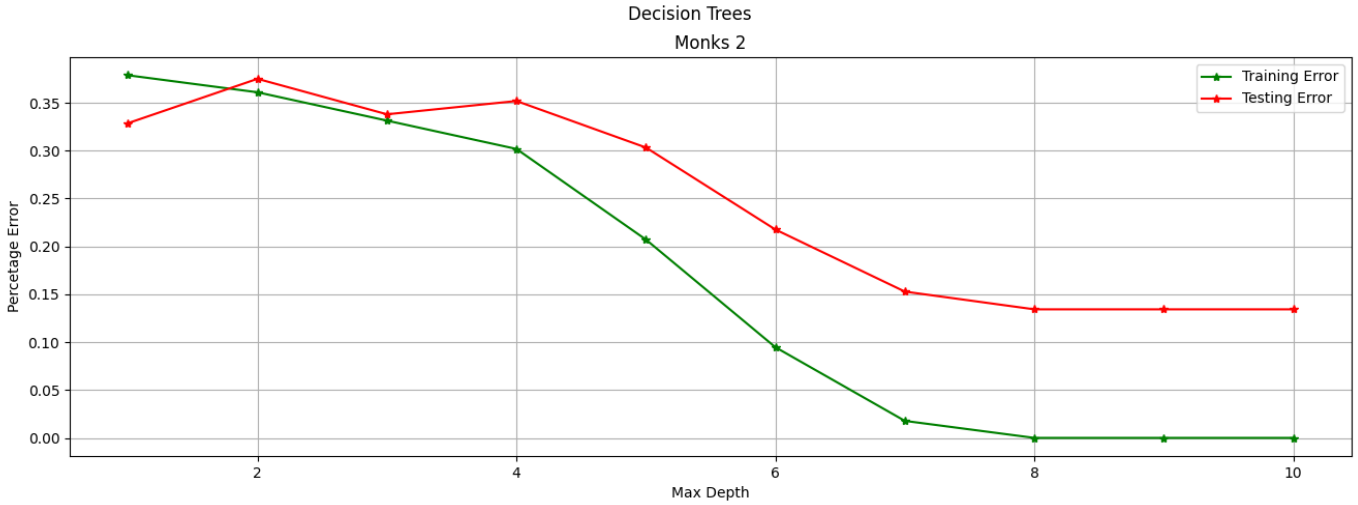
Part – A

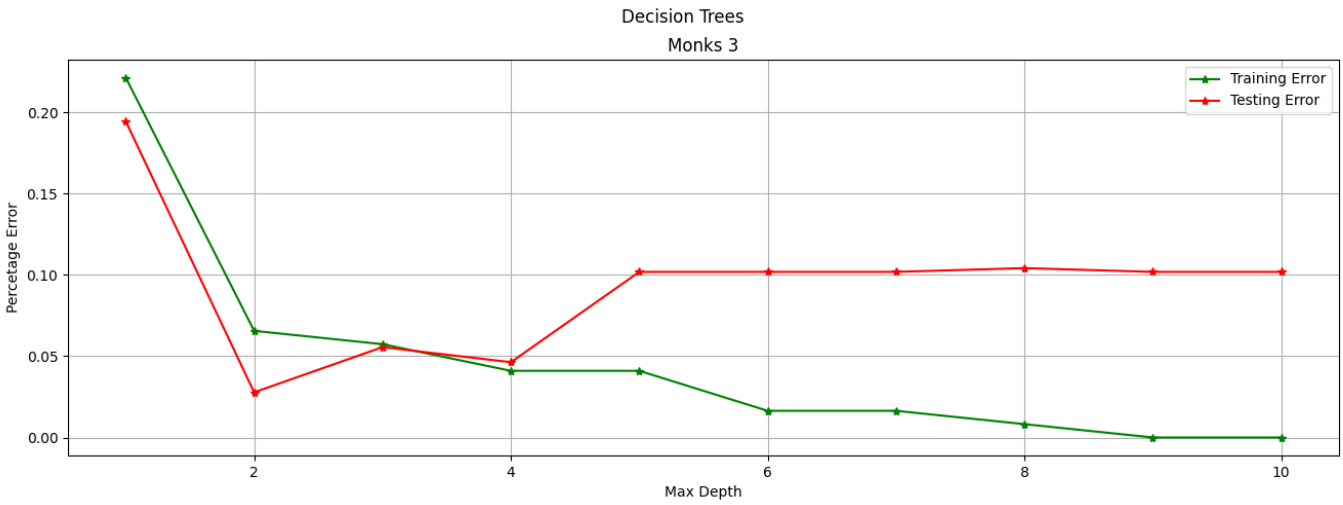
a. (Learning Curves, 40 points) For depth $= 1, \ldots, 10$, learn decision trees and compute the average training and test errors on each of the three MONK's problems. **Make three plots, one for each of the MONK's problem sets**, plotting training and testing error curves together for each problem, with tree depth on the x-axis and error on the y-axis.

As you can see in the plots on the next page, the training and testing error reduce as the depth increases for the datasets 1 and 2. Also, while the training error reaches 0 at depth 10 for those datasets, the testing error plateaus near 8-15% for both sets 1 and 2. On dataset 3 on the other hand, the testing error increases drastically after depth 4, while training error decreases. This shows that there is overfitting happening in the 3rd Monk dataset as depth increases

Plots for Training Error vs Testing Error on different depths for the 3 Monk datasets







```
C:\UTD\Sem 2\Machine Learning\PA1>python decision_tree.py
Train Error for Monks-1 on Depth 1 = 26.61%.
Test Error for Monks-1 on Depth 1 = 25.00%.


Train Error for Monks-1 on Depth 2 = 26.61%.
Test Error for Monks-1 on Depth 2 = 25.00%.


Train Error for Monks-1 on Depth 3 = 8.87%.
Test Error for Monks-1 on Depth 3 = 16.67%.


Train Error for Monks-1 on Depth 4 = 8.87%.
Test Error for Monks-1 on Depth 4 = 16.67%.


Train Error for Monks-1 on Depth 5 = 8.06%.
Test Error for Monks-1 on Depth 5 = 16.67%.


Train Error for Monks-1 on Depth 6 = 4.84%.
Test Error for Monks-1 on Depth 6 = 11.11%.


Train Error for Monks-1 on Depth 7 = 3.23%.
Test Error for Monks-1 on Depth 7 = 9.26%.


Train Error for Monks-1 on Depth 8 = 2.42%.
Test Error for Monks-1 on Depth 8 = 9.26%.


Train Error for Monks-1 on Depth 9 = 1.61%.
Test Error for Monks-1 on Depth 9 = 8.33%.


Train Error for Monks-1 on Depth 10 = 0.00%.
Test Error for Monks-1 on Depth 10 = 7.41%.
```

Figure 1: Displaying the training and testing error on Monk-1 Dataset for different depths

Part – B

b. (**Weak Learners**, 30 points) For `monks-1`, report the **learned decision tree** and the **confusion matrix on the test set** for `depth=1` and `depth=2`. A **confusion matrix** is a table that is used to describe the performance of a classifier on a data set. For binary classification problems, it will be:

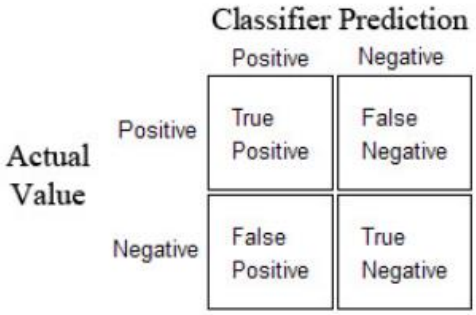


Figure 1: Confusion matrix for a binary classification problem.

Codes were written for displaying binary and multiclass confusion matrices.

```
TREE
+-- [SPLIT: x4 = 1]
|       +-- [LABEL = 0]
+-- [SPLIT: x4 = 1]
|       +-- [LABEL = 1]

                        Predicted Positive      Predicted Negative
Actual Positive         108                     108
Actual Negative         0                       216

Train Error for Monks-1 on Depth 1 = 26.61%.
Test Error for Monks-1 on Depth 1 = 25.00%.
```

Figure 2: Visualization of Tree, Confusion Matrix and Error Rate for Depth 2 of Monk-1 dataset

```
TREE
+-- [SPLIT: x4 = 1]
|       +-- [SPLIT: x0 = 1]
|       |       +-- [LABEL = 0]
|       +-- [SPLIT: x0 = 1]
|       |       +-- [LABEL = 0]
+-- [SPLIT: x4 = 1]
|       +-- [LABEL = 1]

                        Predicted Positive      Predicted Negative
Actual Positive         108                     108
Actual Negative         0                       216

Train Error for Monks-1 on Depth 2 = 26.61%.
Test Error for Monks-1 on Depth 2 = 25.00%.
```

Figure 3: Visualization of Tree, Confusion Matrix and Error Rate for Depth 3 of Monk-1 dataset

Part – C

c. (`scikit-learn`, 15 points) For `monks-1`, use `scikit-learns`'s default decision tree algorithm[2] to learn a decision tree. Visualize the learned decision tree using `graphviz`[3]. Report the **visualized decision tree** and the **confusion matrix** on the test set. **Do not change the default parameters.**

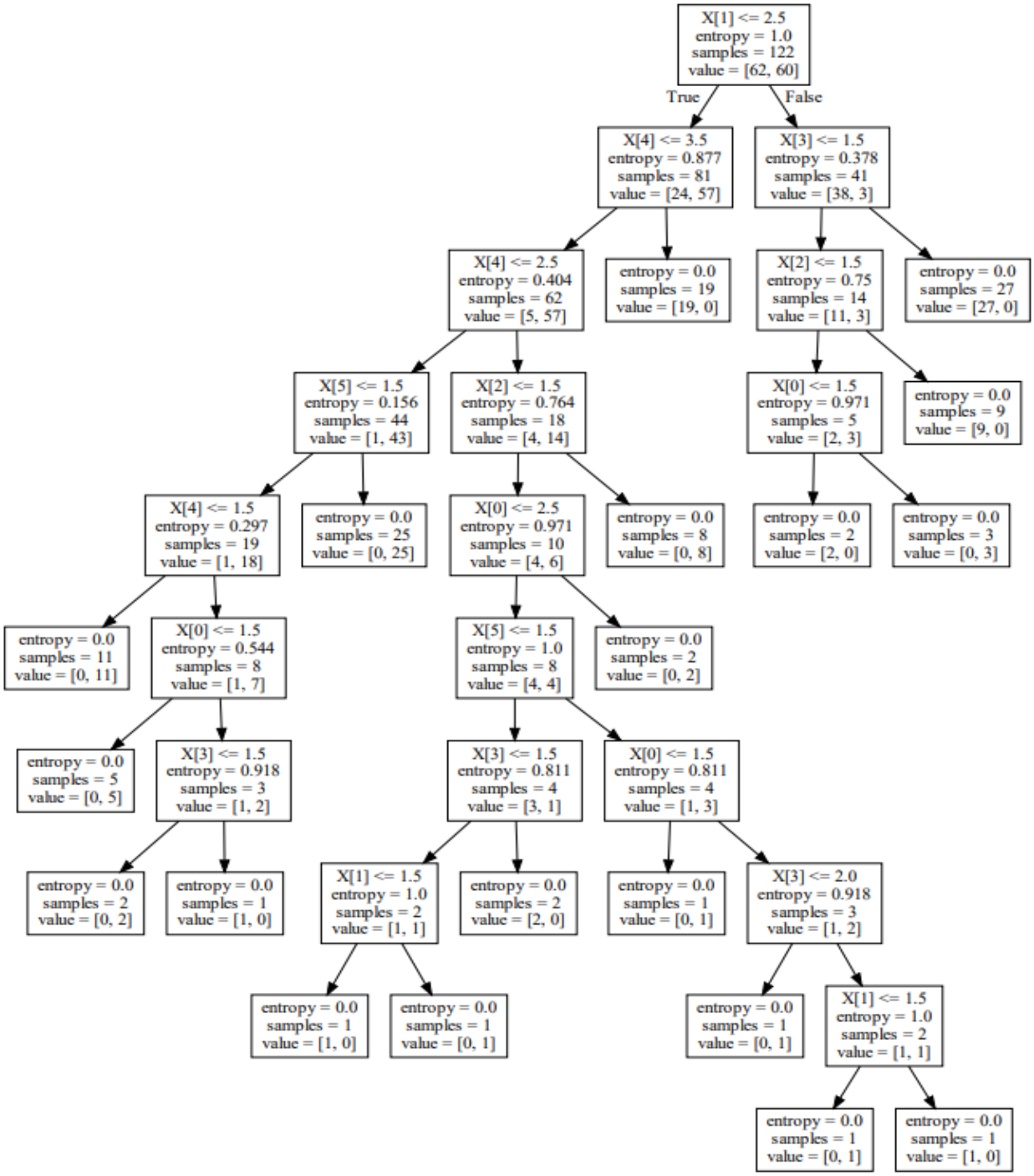


Figure 4: Visualization of the Sklearn Decision Tree on the Monk-1 Dataset made using Graphviz

NOTE: The above graph was made by changing the splitting criterion of the decision tree from 'gini' to entropy. The Decision Tree made from Gini is also present in the submitted solution

```
Confusion Matrix for the Sklearn decision tree on the Monk-1 Dataset


                        Predicted Positive      Predicted Negative
Actual Positive         197                     31
Actual Negative         12                      192
```

Figure 5: Confusion Matrix of the sklearn Decision Tree on the Monk-1 Dataset

Part – D

d. (**Other Data Sets**, 15 points) Repeat steps 2 and 3 with your "own" data set and report the confusion matrices. You can use other data sets in the UCI repository. If you encounter continuous features, consider a simple discretization strategy to pre-process them into binary features using the mean. For example, a continuous feature x can be discretized using its mean μ as

$$x_{\text{binary}} = \begin{cases} 0, & \text{if } x \leq \mu, \\ 1, & \text{if } x > \mu. \end{cases}$$

For Part D of the Programming Assignment, I opted to use the Iris Dataset which classifies the type of iris plant based on attributes of its sepal and petal. There are 4 features and 3 classes of iris. The features were discretized by converting their decimal values into integer data. Also. the classes were encoded in order to model the decision tree using our id3 algorithm. Hence the three classes were made into integers (0, 1, 2). The split between training and testing was 80-20

Here are the results from applying the ID3 algorithm and Sklearn's Decision tree on the iris Dataset

```
Depth 1 Decision Tree on Iris Dataset
TREE
+-- [SPLIT: x3 = 0]
|       +-- [LABEL = 1]
+-- [SPLIT: x3 = 0]
|       +-- [LABEL = 0]

                  Pred Class 0            Pred Class 1            Pred Class 2
Actual Class 0    10                      0                       0
Actual Class 1    0                       9                       0
Actual Class 2    0                       11                      0

Train Error for Iris on Depth 1 = 33.33%.
Test Error for Iris on Depth 1 = 36.67%.
```

Figure 6: Visualization of Tree, Confusion Matrix and Error Rate for Depth 1 of Iris Dataset

```
Sklearn Decision Tree on Iris Dataset

                  Pred Class 0            Pred Class 1            Pred Class 2
Actual Class 0    10                      0                       0
Actual Class 1    0                       7                       2
Actual Class 2    0                       2                       9

Train Error for Iris using Sklearn = 4.17%.
Test Error for Iris using Sklearn = 13.33%.
```

Figure 7: Visualization of Tree, Confusion Matrix and Error Rate for Depth 2 of Iris Dataset

```
Depth 2 Decision Tree on Iris Dataset
TREE
+-- [SPLIT: x3 = 0]
|       +-- [SPLIT: x3 = 1]
|       |       +-- [LABEL = 2]
|       +-- [SPLIT: x3 = 1]
|       |       +-- [LABEL = 1]
+-- [SPLIT: x3 = 0]
|       +-- [LABEL = 0]

                  Pred Class 0            Pred Class 1            Pred Class 2
Actual Class 0    10                      0                       0
Actual Class 1    0                       6                       3
Actual Class 2    0                       0                       11

Train Error for Iris on Depth 2 = 11.67%.
Test Error for Iris on Depth 2 = 10.00%.
```

Figure 8: Confusion Matrix and Error Rate for Sklearn's Decision Tree

NOTE: The below tree was made by changing the splitting criterion of the decision tree from 'gini' to entropy. The Decision Tree made from Gini is also present in the submitted solution

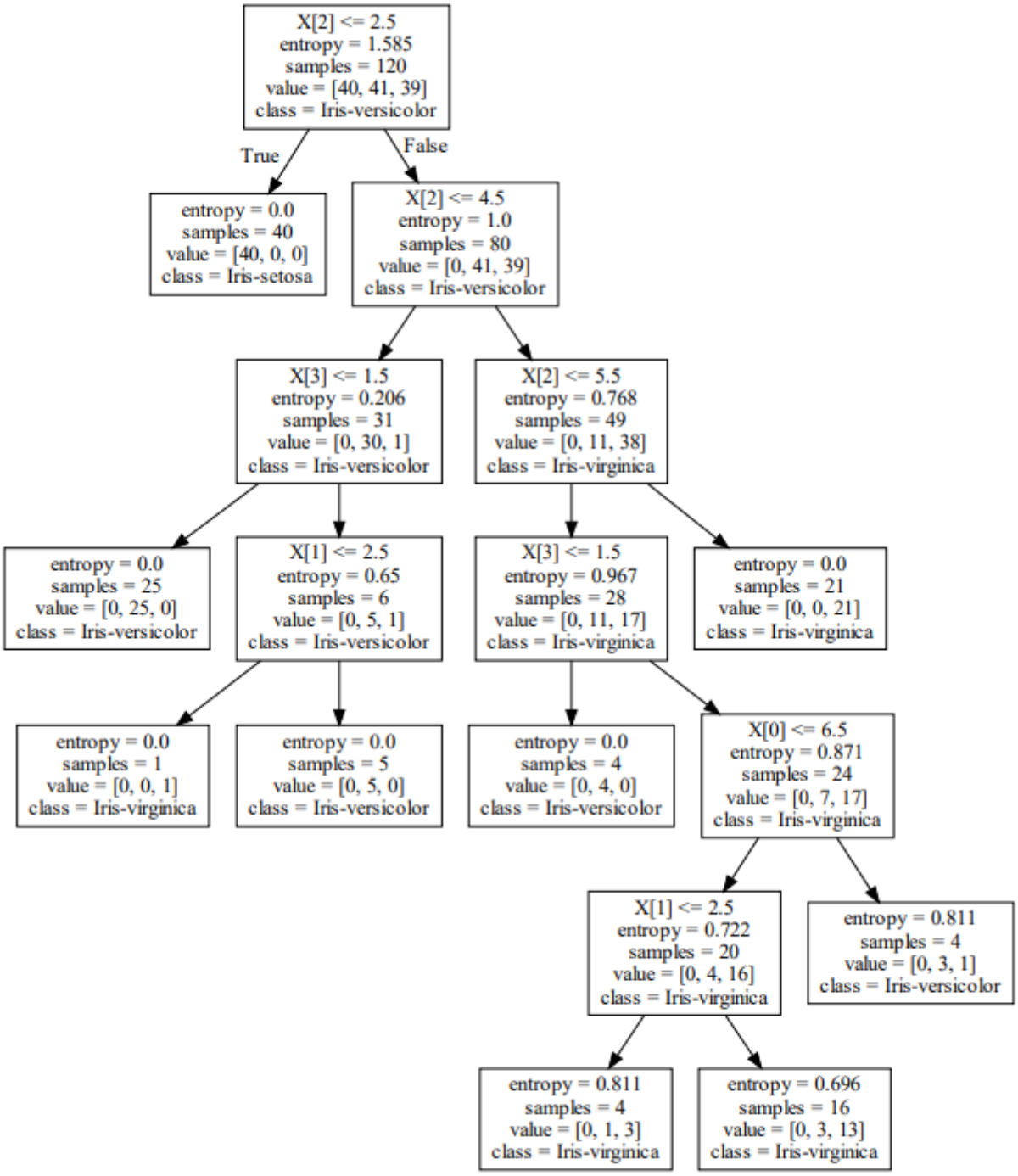


Figure 9: Visualization of the Sklearn Decision Tree on the Iris Dataset made using Graphviz