CS 6375: Machine Learning
Assignment #2

Prof. Anjum Chida

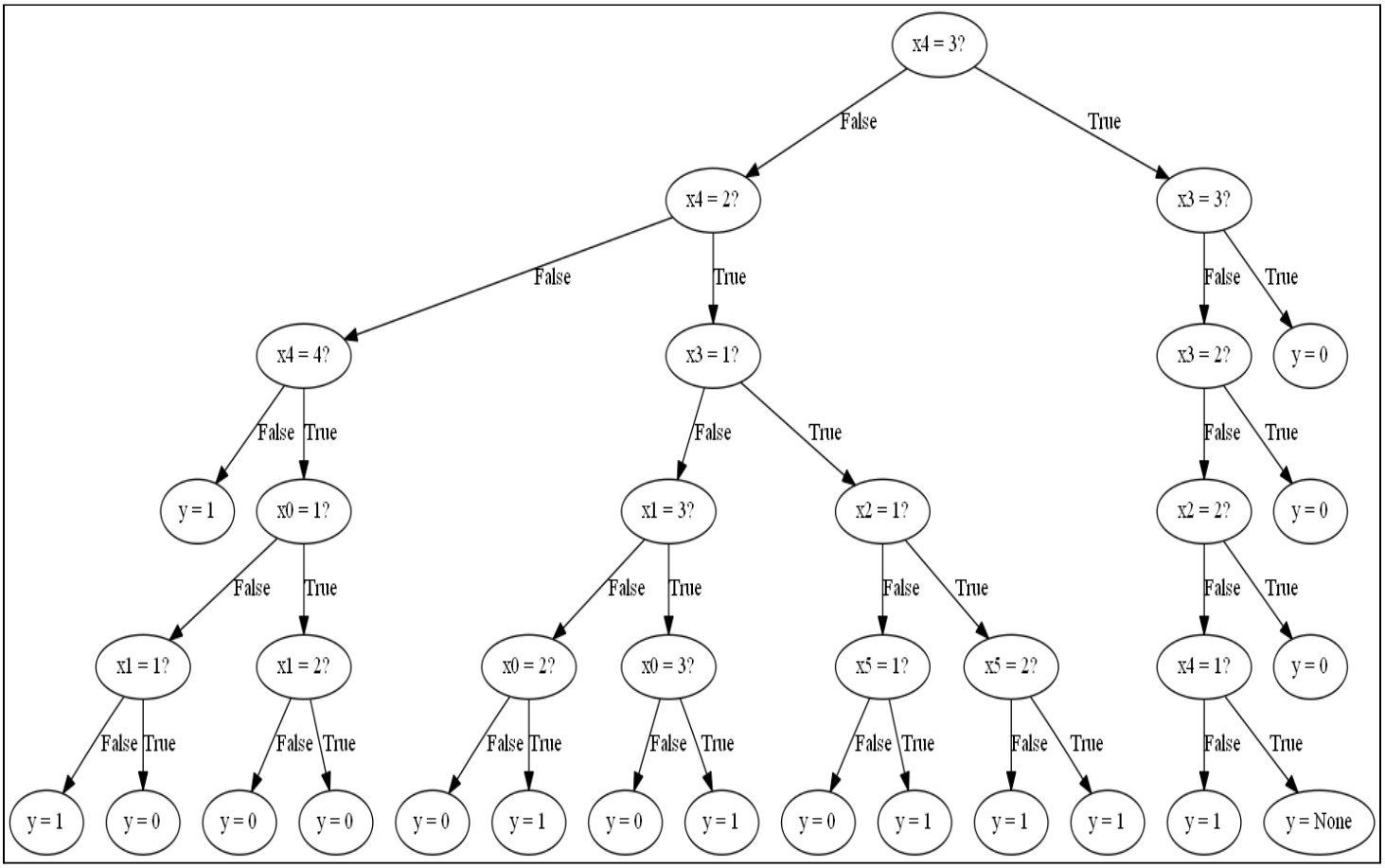
Name: Akash Biswal

NetID: axb200166

a)

The general python code for the ID3 algorithm was implemented and shown below are the results for:

- Dataset: monks-1.train, monks-1.test
- Parameter: max_depth = 3

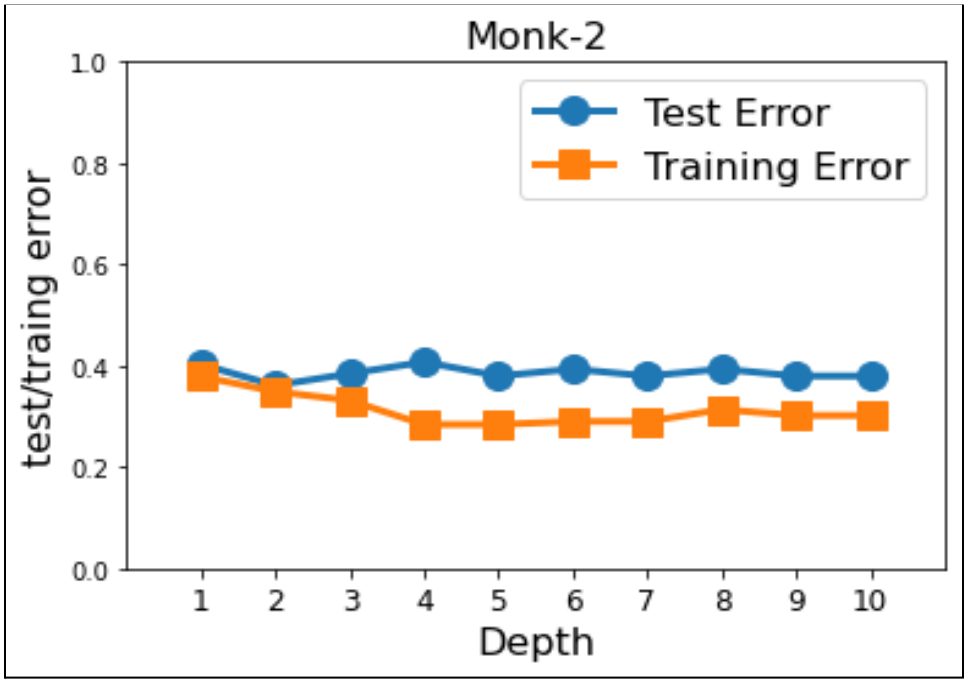


Training Error = 17.74%.
Test Error = 26.62%.

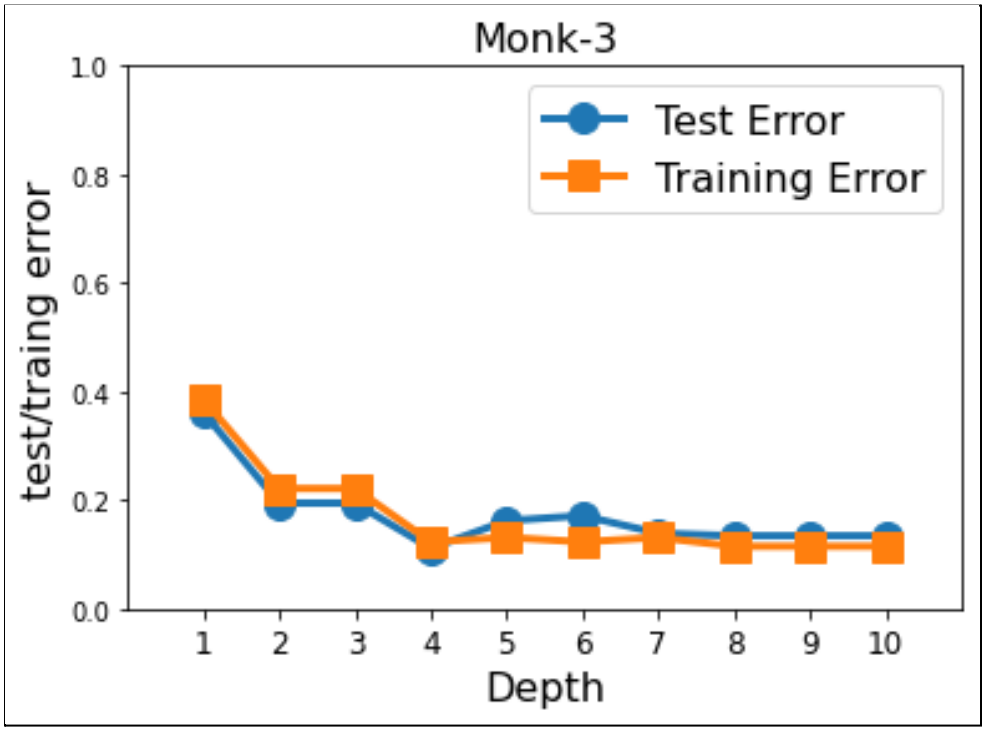
b) The decision trees were learned for all depths from 1-10 for each of the monk datasets

Dataset: monks-1.train, monks-1.test

Dataset: monks-2.train, monks-2.test



Dataset: monks-3.train, monks-3.test

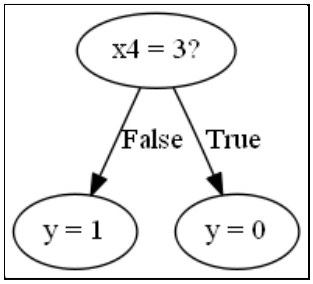


For the monk-1 dataset/problem at depths (4,5,6) the difference between training and test error is high which hints at overfitting. For monk-2 problem after depth 4 the difference between training and test error remains somewhat constant which displays a poorly performing model. The results are best for monk-3 as there is very miniscule training and test error difference at all depths. This problem was solved best by the ID3 algorithm.

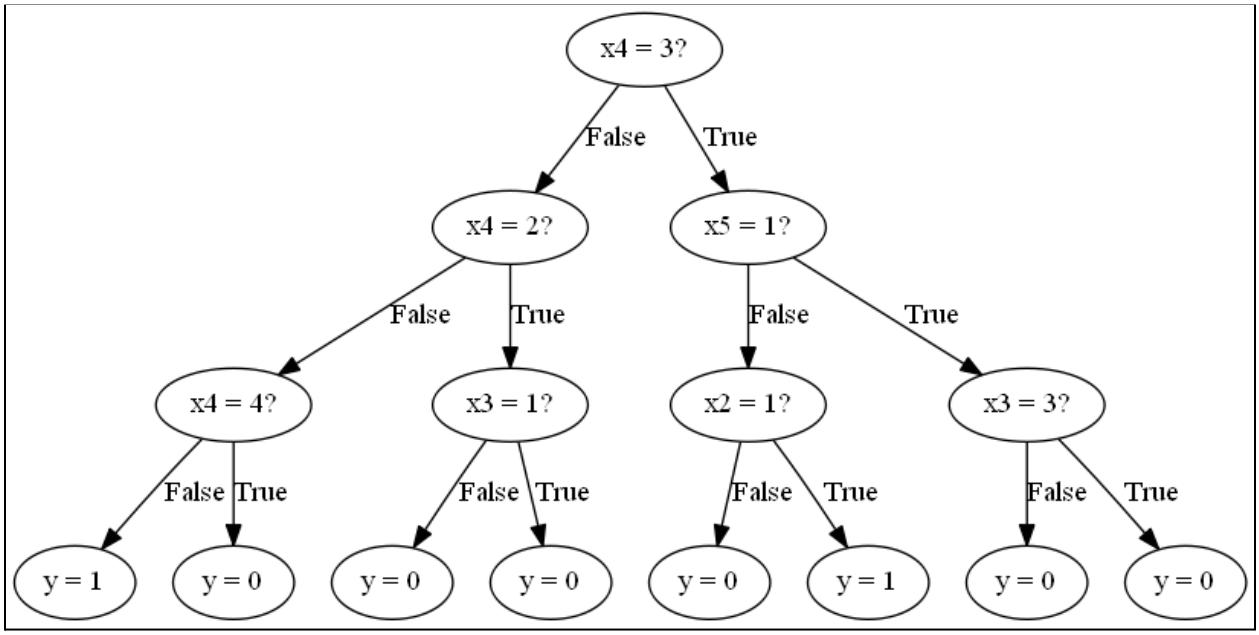
---

c)

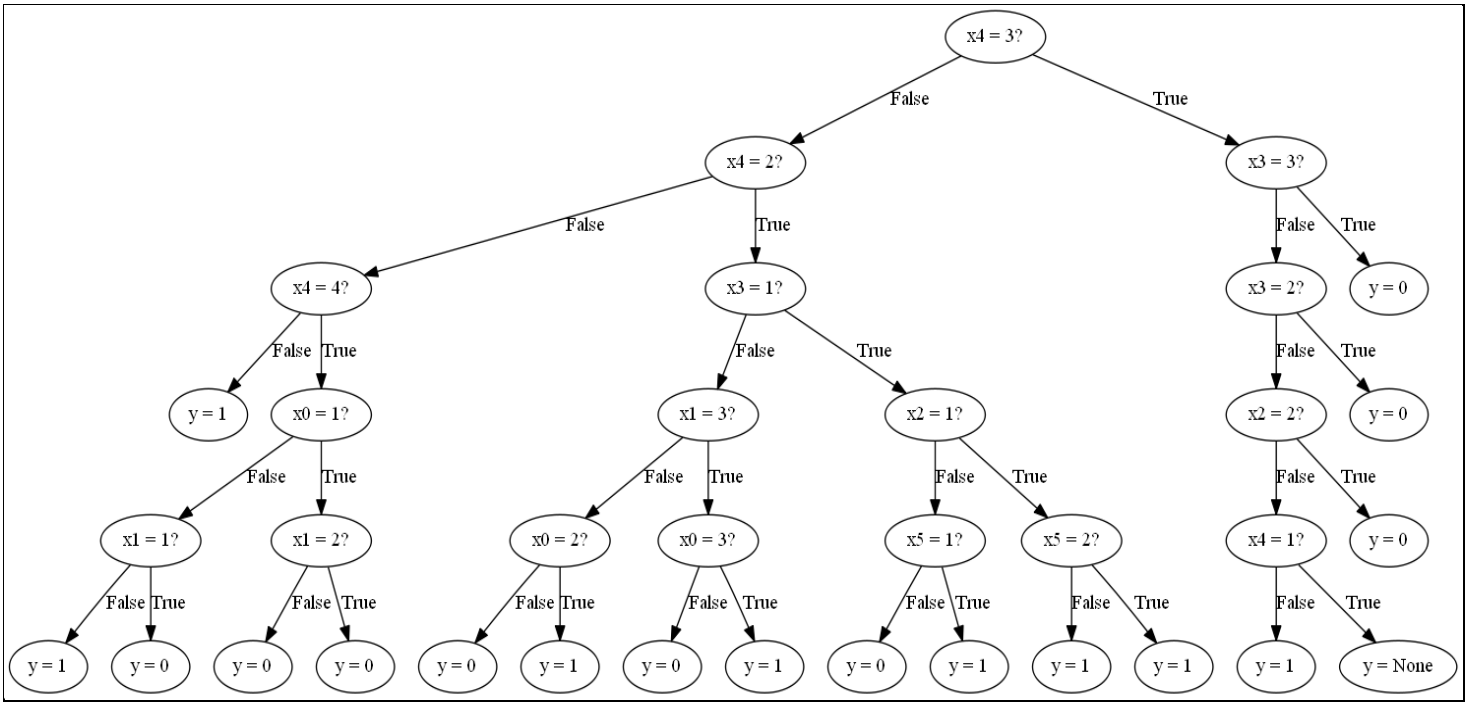
Decision trees and confusion matrices for depth 1,3,5 learnt using the ID3 algorithm on monk-1



tree depth:1 confusion matrix
[[72 144]
[36 180]]



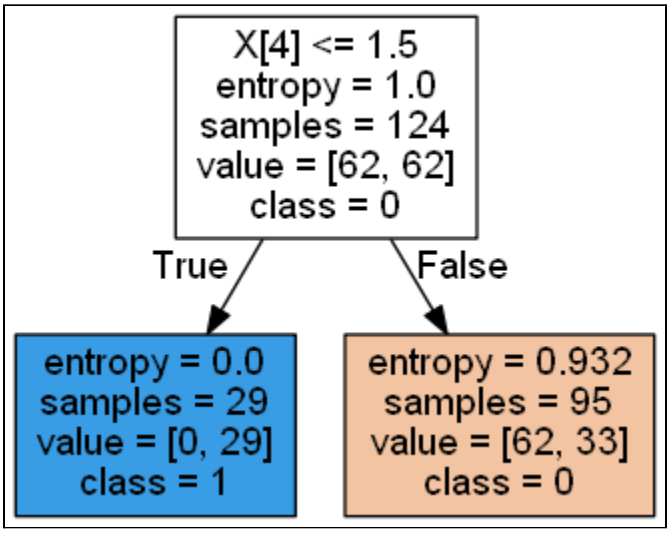
tree depth:3 confusion matrix
[[198 18]
[99 117]]



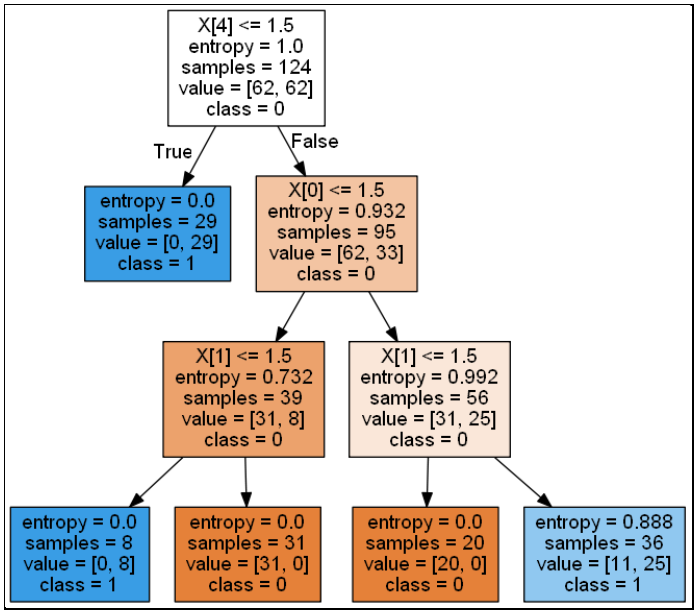
tree depth:5 confusion matrix
[[154 62]
[53 163]]

---

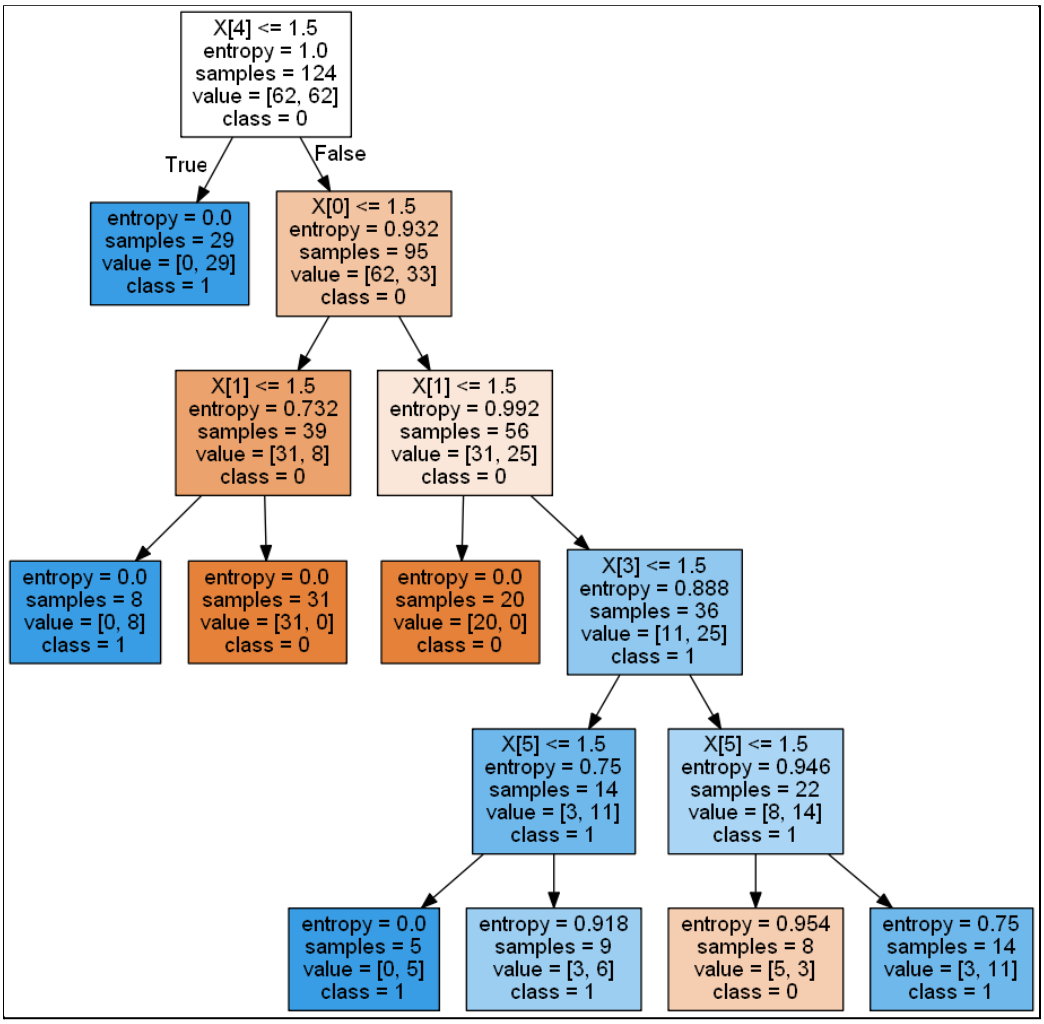
d) Using the scikit-learn DecisionTreeClassifier, decision trees were learnt on the monk-1 problem and their trees and confusion matrices were plotted



scikit tree depth:1 confusion matrix
[[216 0]
[108 108]]



scikit tree depth:3 confusion matrix
[[144 72]
[0 216]]



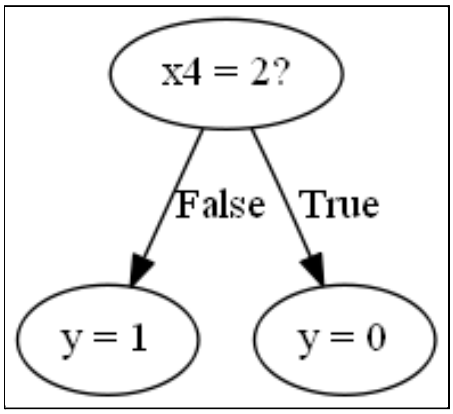
scikit tree depth:5
confusion matrix
[[168 48]
[24 192]]

e)
For this part of the assignment the Tic-Tac-Toe Endgame Data Set from the UCI repository was used. This database encodes the complete set of possible board configurations at the end of tic-tac-toe games, where "x" is assumed to have played first. The target concept is "win for x" (i.e., true when "x" has one of 8 possible ways to create a "three-in-a-row"). The classes are {positive, negative} where positive signifies wins for player “x”.

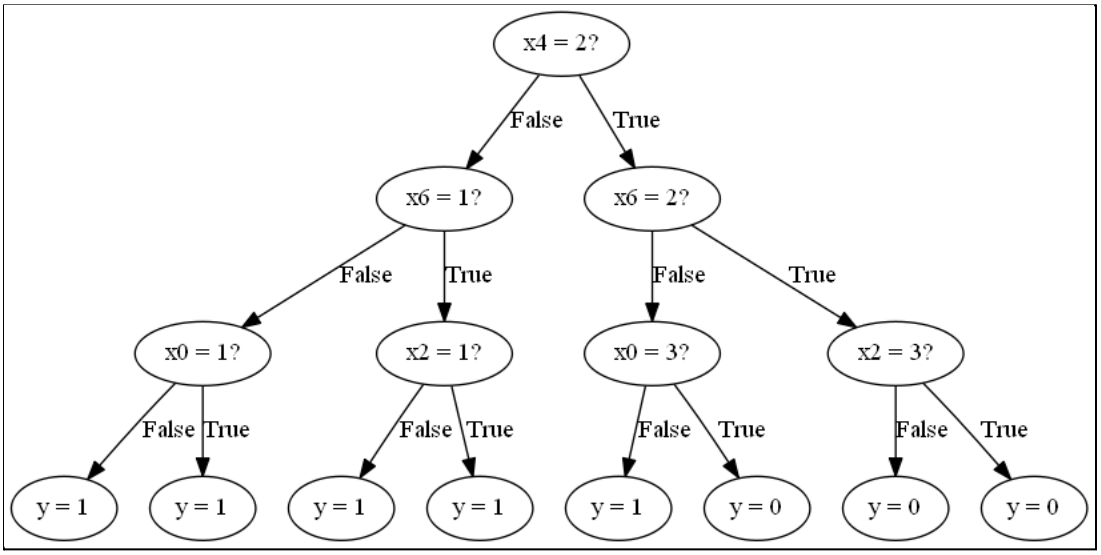
Number of Instances: 958 (legal tic-tac-toe endgame boards)
Number of Attributes: 9, each corresponding to one tic-tac-toe square
Labels: {positive, negative}, column 10
Attribute Information: (x=player x has taken, o=player o has taken, b=blank)

Preprocessing:
Initially the dataset had values in string format and it was preprocessed to replace these strings with numbers
x - 1; o - 2; b - 3; positive - 1; negative - 0

Results(tree+confusion matrix) with ID3 algorithm for depth 1,3,5:



tictactoe id3 tree depth:1 confusion matrix
[[45 26]
[31 90]]
Training Error = 30.16%.
Test Error = 29.69%.



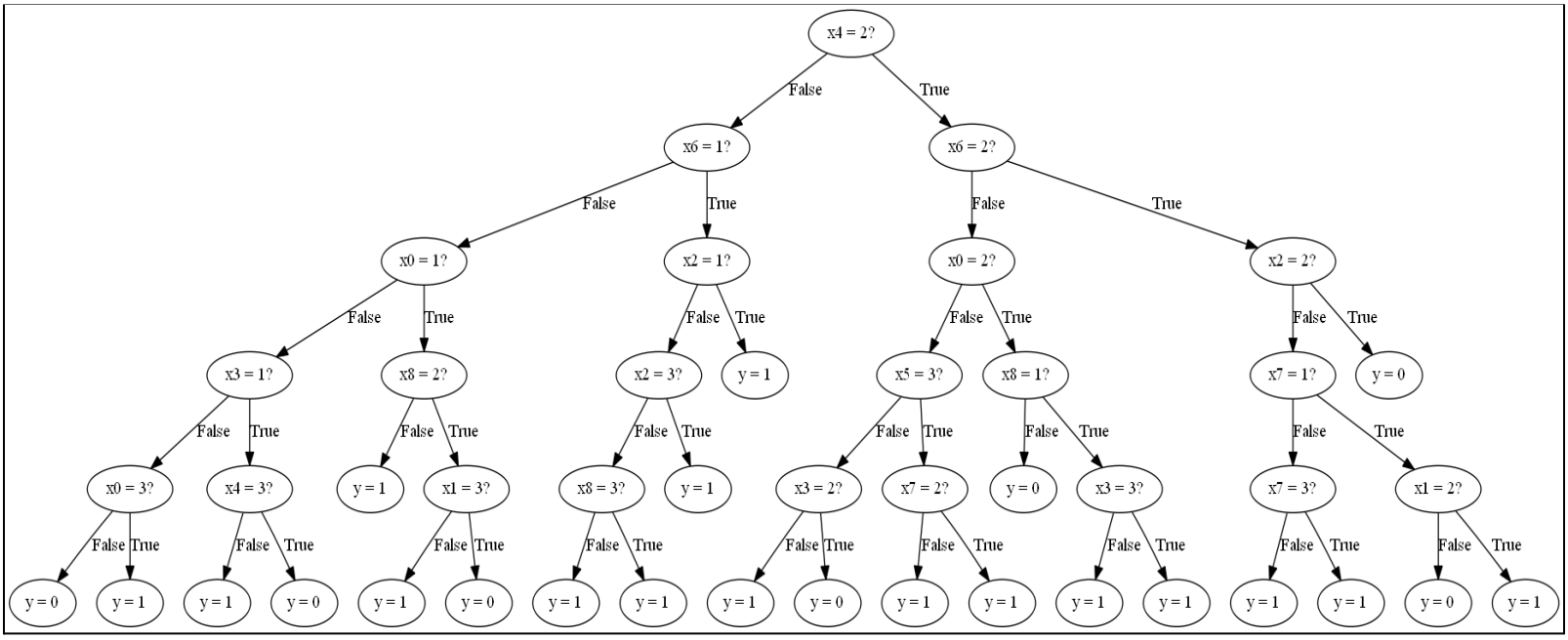
tictactoe id3 tree depth:3 confusion matrix

[[18 53]

[15 106]]

Training Error = 29.24%.

Test Error = 35.42%.

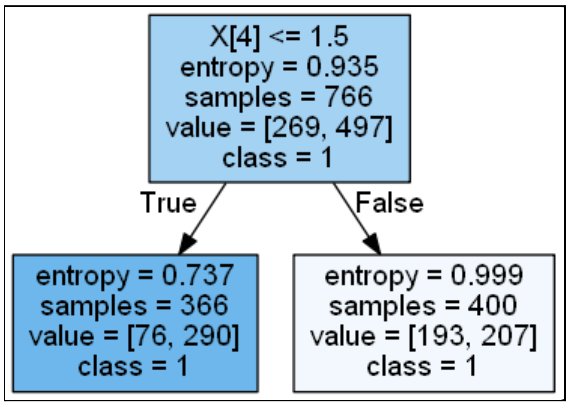


tictactoe id3 tree depth:5 confusion matrix

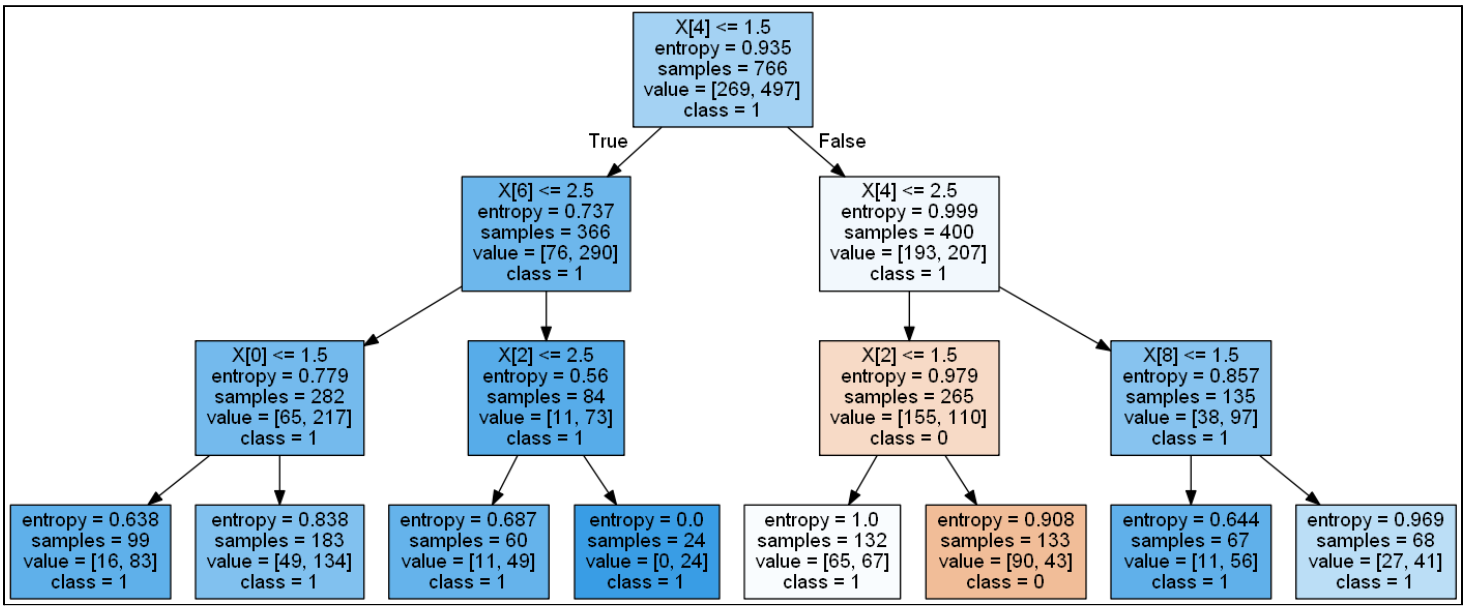
[[45 26]

[14 107]]

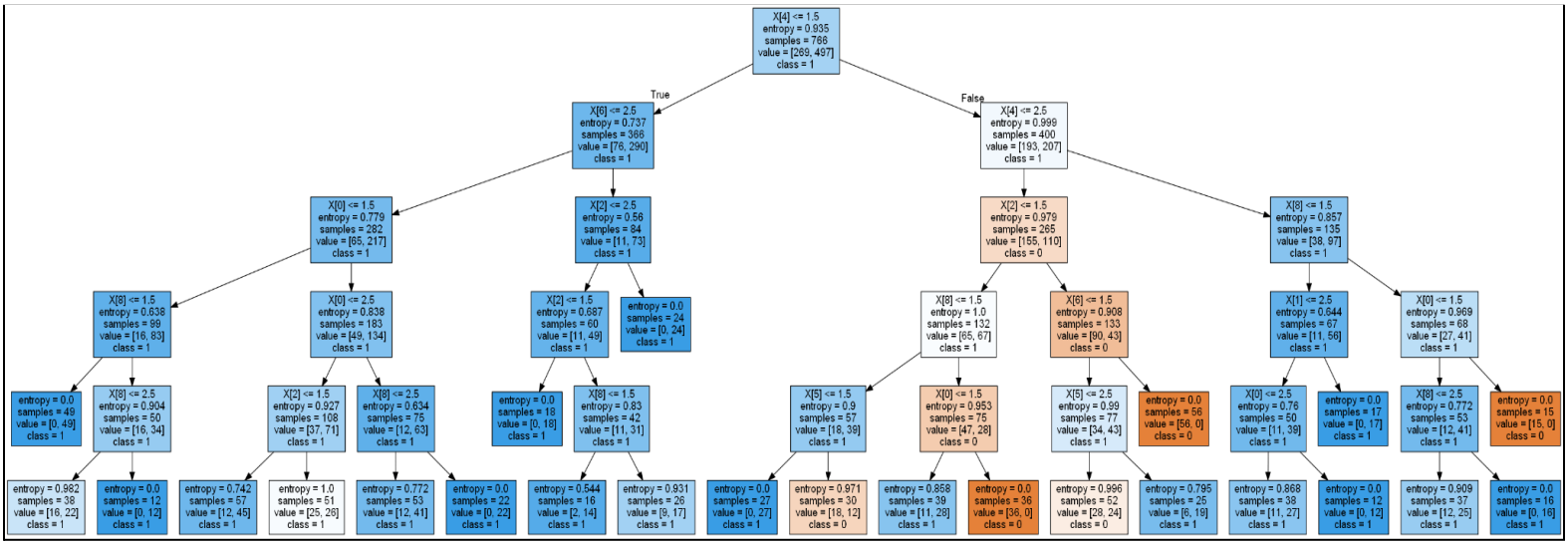
Training Error = 19.06%.
Test Error = 20.83%.
Results(tree+confusion matrix) with sklearn DecisionTreeClassifier algorithm for depth 1,3,5:



tictactoe scikit tree depth:1 matrix
[[0 63]
[0 129]]
Training Error = 35.12%.
Test Error = 32.81%.



tictactoe scikit tree depth:3 confusion matrix
[[21 42]
[18 111]]
Training Error = 28.98%.
Test Error = 31.25%.



tictactoe scikit tree depth:5 confusion matrix

[[39 24]

[12 117]]

Training Error = 19.84%.

Test Error = 18.75%.

The ID3 performs better on the tictactoe dataset at lower depths but it is seen that when the depths are increased the sklearn DecisionTreeClassifier outperforms the ID3 algorithm.
The sklearn algorithm is in general better performing than ID3.