Homework #3

The **coding component** of this assignment should be **submitted through e-Learning** before **12:59 pm Central Daylight Time on Wednesday, March 27, 2019**. The **report component** of this assignment is due **at the start of class** on **Wednesday, March 27, 2019**. These deadlines are without exceptions unless permission was obtained from the instructor **in advance**.
You may collaborate with other students, discuss the problems and work through solutions together. However, you **must write up your code and solutions on your own**, without copying another student's work or letting another student copy your work. In your solution for each problem, you must write down the names of any person with whom you discussed it. This will **not** affect your grade.

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 MONK's 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), and binary class labels (column 1). See `monks.names` for more details.

Visualization: The code skeleton provided contains a function `render_dot_file()`, which can be used to generate `.png` images of the trees learned by both `scikit-learn` and your code. See the documentation for `render_dot_file()` for additional details on usage.

a. (**Autograder Score**, 20 points) Your code will be auto-graded and cross-checked with other submissions. The autograder will evaluate your code on several different data sets to perform a sanity check. In order to ensure that your code passes the autograder, 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.

b. (**Learning Curves**, 20 points) For depth = 1, . . . , 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.

c. (**Weak Learners**, 20 points) For `monks-1`, report the visualized **learned decision tree** and the **confusion matrix on the test set** for depth = 1, 3, 5. You may use `scikit-learns`'s `confusion_matrix()` function[2].

d. (`scikit-learn`, 20 points) For `monks-1`, use `scikit-learn`'s `DecisionTreeClassifier`[3] to learn a decision tree using `criterion='entropy'` for depth = 1, 3, 5. You may use `scikit-learn`'s `confusion_matrix()` function[4].

e. (**Other Data Sets**, 20 points) Repeat steps (c) and (d) 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

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

Write a **report** with the solutions to the questions above, showing the plots, confusion matrices and a **brief discussion** (4–5 lines) comparing your implementation to that of `scikit-learn`, which is a widely-used, publicly-available, open-source implementation.

[1] https://archive.ics.uci.edu/ml/datasets/MONK's+Problems
[2] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html
[3] http://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html
[4] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html