Project 1: Decision Trees for Breast Cancer Diagnosis
COSC425, Fall 2020

Due: Sept 21 @ 11:59pm

In this project, you will be implementing a decision tree solution for breast cancer diagnosis. This assignment follows the specification in the "Project Guidelines" document on Canvas.

I. General Information

Your project is responsible for classifying instances as benign or malignant. Your report should cover the relevant steps in the "Project Workflow", reporting what you did and the results. You are responsible for implementing your own Decision Tree functions, with the exception of basic numerical libraries.

II. Dataset

This project will use the Breast Cancer Wisconsin dataset from the UCI Machine Learning repository. The dataset file is named `breast-cancer-wisconsin.data`, and is included in the `project1-starter.zip` file on Canvas. The data includes the following features:

1. Sample code number: id number (**Note**: This is not a feature.)
2. Clump Thickness: 1–10
3. Uniformity of Cell Size: 1–10
4. Uniformity of Cell Shape: 1–10
5. Marginal Adhesion: 1–10
6. Single Epithelial Cell Size: 1–10
7. Bare Nuclei: 1–10
8. Bland Chromatin: 1–10
9. Normal Nucleoli: 1–10
10. Mitoses: 1–10
11. Class (2 is Benign, 4 is Malignant)

III. Implementation Requirements

You are tasked with implementing a univariate decision tree classifier (i.e., a classifier that evaluates only one feature at each node). Your implementation for this classifier has two constraints:

1. You must implement a **`construct()`** function that creates a decision tree from a provided dataset (e.g. training set). This function should dynamically construct the tree by calculating the information gain of prospective splits based on the given data. This function should include an optional parameter that allows you to threshold the tree based on (1) an information gain value or (2) a maximum tree depth.
2. You must implement a **`classify()`** function that takes the constructed decision tree and applies it to a different dataset (i.e., test set). This function should return a array (i.e., a list in Python) of predicted classifications that are the product of your decision tree. The function should return the depth of the tree after construction has been completed.

IV. Starting Code
For many of you, this may be your first time using Python. To ease the burden of determining how to structure your code, you're being provided with a starter file named "main.py" that implements and uses a `DecisionTreeBuilder` class, alongside two other classes – `InternalNode` and `LeafNode`. There is still much to implement, which is why you are being given a head-start!

You can run the file by downloading / installing Python 3.7 on your machine, and running 'python main.py' in your Terminal. The file can be found on Canvas.

III. Measuring Performance
A central part of your final report involves evaluating the performance of your learning algorithm. To do this, you will need to compare the correct classification of the input data with your predicted classifications. You should therefore implement a function that prints simple performance metrics about your classification. This includes `TP` = the number of true positives, `TN` = the number of true negatives, `FP` = the number of false positives, and `FN` = the number of false negatives. Your function should specifically report these in the form of a **Confusion Matrix** formatted as follows:

	Predicted Class	
True Class	Benign	Malignant
Benign	`TN`	`FP`
Malignant	`FN`	`TP`

In addition to the Confusion Matrix, you should also print out several other metrics:

- Accuracy = `(TN + TP) / (TN + TP + FN + FP)`
- Recall = `TP / (TP + FN)`
- Precision = `TP / (TP + FP)`
- F1 Score = `2 x Precision x Recall / (Precision + Recall)`

IV. Evaluating the Decision Tree
The goal of your final report is to communicate the strengths of your learning approach. You should apply your decision tree functions to the breast cancer training data with a variety of maximum depths (i.e. k=1 to 10). In parallel, you should explore the effectiveness of using different information gain thresholds. To provide reliability in the observations you make about these hyperparameters, you should employ cross-validation. In your report, you should detail the combinations of these hyperparameters that you explored. As a final step, use the classifier with these hyperparameters on your test data and report the performance you receive.

V. Writing the Report
The "Project Guidelines" document outlines the general format that you should use for your format and the grading rubric used to evaluate your submission. Note that the report is a requirement for project submission.