CS 6362: Machine Learning
Assignment 1

Due: Wednesday, 09/18/2019, at midnight

General Instructions:

If anything is ambiguous or unclear:

1. **Discuss possible interpretations with other students, your TA, and instructor. You may email your instructor or TA if you have specific questions or need clarification on questions.**
2. **Make assumptions, state them explicitly, and then use them to work out the problem**

Remember that after general discussions with others, you are required to work out the solutions by yourself. Do not plagiarize code from any available source. All submitted work must be your own work. Please refer to the Honor code for clarifications.

One very interesting application area of machine learning is in making medical diagnoses. In this problem, you will train and test a binary decision tree to detect breast cancer using real world data. *You may use any programming language you like*, but Python is the preferred language. A library may be used for the underlying tree data structure. In Python, one such library is "anytree".

We will use the Wisconsin Diagnostic Breast Cancer (WDBC) dataset, which you can download from http://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic) . The dataset consists of 569 samples of biopsied tissue. The tissue for each sample is imaged and 10 characteristics of the nuclei of cells present in each image are characterized. These characteristics are:

1. Radius
2. Texture
3. Perimeter
4. Area
5. Smoothness
6. Compactness
7. Concavity
8. Number of concave portions of contour
9. Symmetry
10. Fractal dimension

Each of the 569 samples used in the dataset consists of a feature vector of length 30. The first 10 entries in this feature vector are the mean of the characteristics listed above for each

image. The second 10 are the standard deviation and last 10 are the largest value of each of these characteristics present in each image.

Each sample is also associated with a label: a label of value 1 indicates the sample was for malignant (cancerous) tissue. A label of value 0 indicates the sample was for benign tissue.

Learning a Binary Decision Tree. As discussed in class, to learn a binary decision tree we must determine which feature (attribute) to select as well as the threshold value to use in the split criterion for each non-leaf node in the tree. This can be done in a recursive manner, where we first find the optimal split for the root node using all of the training data available to us. We then split the training data according to the criterion selected for the root node, which will leave us with two subsets of the original training data. We then find the optimal split for each of these subsets of data, which gives the criterion for splitting on the second level child nodes. We recursively continue this process until the subsets of training data we are left with at a set of children nodes are pure (i.e., they contain only training examples of one class) or the feature vectors associated with a node are all identical (in which case we cannot split them) but their labels are different.

In this problem, you will implement an algorithm, coded as *computeOptimalSplit*, to learn the structure of a tree. The optimal splits at each node should be found using the gain computed from the: (1) weighted Gini index, and (2) the information gain (entropy) ratio measure discussed in class. Please write modular code so the Impurity measure is a function that is called from the *computeOptimalSplit* function.

Please provide a brief write up of your splitting algorithm as documentation for how you implemented this method. This documentation can be embedded in your code, but please provide a summary and pseudocode as a separate document.

Notes: (1) While there are multiple ways to design a decision tree, in this problem you will constrain yourself to ones that simply pick one feature to split on. Further, you will restrict yourself to performing only binary splits. In other words, each split should simply determine if the value of a particular feature in the feature vector of a sample is less than or equal to a threshold value or greater than the threshold value.

(2) Please note that the feature values in the provided dataset are continuous-valued.

Pruning a Binary Decision Tree. The method of learning the structure and splitting criterion for a binary decision tree described above terminates when the training examples associated with a node are all of the same class or there are no more splits that are possible. In general, this will lead to overfitting. As discussed in class, pruning using a validation set is one way to avoid overfitting.

In this problem, you will implement an algorithm to use validation data to greedily prune a binary decision tree in an iterative manner. Specifically, the algorithm that we will implement will start with a binary decision tree and perform an exhaustive search for the single node for which removing it (and its children) produces the largest increase (or smallest decrease) in

classification accuracy as measured using validation data. Once this node is identified, it and its children are removed from the tree, producing a new tree. This process is repeated, where we iteratively prune one node at a time until we are left with a tree that consists only the root node[1].

In this problem, you will implement a function, coded as *pruneSingleGreedyNode*, which starts with a tree and selects the single best node to remove to produce the greatest increase (or smallest decrease) in classification accuracy as measured with validation data. As a suggestion, write a function that will return a listing of all possible trees that can be formed by removing a single node from a base tree; and another, which will classify a set of samples given a decision tree.

Again, along with your code please provide documentation built into the code, as well as a brief write up of your pruning algorithm for how you implemented this method. You will submit this documentation along with your code.

Data Analysis

Training a Binary Decision Tree. In this section, you will use the code that you have written above. You will start by training a basic decision tree. For training, we are providing you with two data sets, data set 1, uses 80% of the data for training and 20% for test. Data set 2 uses 90% of the data for training and 10% for test. [The training and test sets will be uploaded on Brightspace by the TA, Azhar Molla.]

Please specify the total number of nodes and the total number of leaf nodes in the tree. Also, please report the classification accuracy (percent correct) of the learned decision tree on the provided training and testing data. Discuss your results.

Pruning a Binary Decision Tree. Now you will make use of the pruning code you have written. Please start with the tree that was just trained in the previous part of the problem and make use of the validation data to iteratively remove nodes in the greedy manner described above. Please continue iterations until a degenerate tree with only a single root node remains. For each tree that is produced, please calculate the classification accuracy for that tree on the training, validation and testing datasets.

After collecting this data, please plot a line graph relating (1) classification accuracy and (2) the F1 score on the test set to the number of leaf nodes in each tree (so number of leaf nodes should be on the X-axis and classification accuracy and F1 score should be on the Y-axis). Please add to this same figure, similar plots for percent accuracy on training and validation data. The number of leaf nodes should range from 1 (for the degenerate tree) to the number present in the unpruned tree. The Y-axis should be scaled between 0 and 1.

[1] In practice, you can often simply continue the pruning process until the validation error fails to increase by a predefined amount. However, for illustration purposes, we will continue until there is only one node left in the tree.

Similarly, also generate the precision-recall and the ROC curves for the same data generated for each of the pruned decision trees. Follow the specifications for these plots as discussed in class.

Please comment on what you notice and how this illustrates overfitting. Include the produced figure and any code you needed to write to produce the figure and calculate intermediate results with your assignment submission.

Discuss the best decision tree proposed by each of these measures.

Drawing a Binary Decision Tree. One of the benefits of decision trees is humans easily understand the classification scheme they encode. Please select the binary decision tree from the pruning analysis above that produced the highest accuracy on the validation dataset and diagram it. (In the event that two trees have the same accuracy on validation data, select the tree with the smaller number of leaf nodes). Do the same for the tree that has the best F1 score.

When stating the feature attributes that are used in splits, please use the attribute names (instead of index) listed in the dataset section of this problem.

Hint: The best decision tree as measured on validation data for this problem should not be too complicated, so if drawing this tree seems like a lot of work, then something may be wrong.

You may use available packages of your choice for generating the decision tree plots, as well as the classification accuracy, F1 score, precision-recall and the ROC plots. The previously mentioned python library "anytree" can create a simple plot of the decision tree with its print() function, but for the other plots you will need separate libraries like https://plot.ly/python/ or https://matplotlib.org/. It is also worth noting that anytree's simple, text-based plot of the decision tree is far from ideal – you are welcome to create a better plot of the tree. Another, more powerful package is "gnuplot.py" (http://gnuplot-py.sourceforge.net/).