Joshua Matni
Project 1: Decision Trees
Write up

DecisionTreeNode Class

1. __init__ Method: This is a class used for initializing a node in the DecisionTree for internal nodes: it saves the feature index on which it should split, the threshold value, and references to the left and right child nodes. And for the leaf nodes, it saves the class value representing the prediction.
2. is_leaf_node Method: Checks if the node is a leaf node. It returns True if the node contains a class value, self.value is not None, meaning it does not have any child nodes and it is a final prediction.

DecisionTree Class

1. __init__ Method: Initializes the Decision Tree Classifier with optional parameters min_samples_split and max_depth. Default values are None for both. min_samples_split is the minimum number of samples required to split an internal node and max_depth is the maximum depth the tree can grow to. The root of the tree self.root is initialized to None.

2. _entropy Method: The entropy of the given set of class labels y: Entropy is a measure of impurity or randomness; it gives quantitative information about how mixed the classes are in the dataset. Lower entropy reflects the fact that the samples are more homogeneous with respect to their class labels.

3. _information_gain Method: This calculates information gain from splitting the dataset on that one feature at this threshold. Information gain is the difference between the entropy before the split-what we'll call "parent entropy"-and the weighted average entropy after the split, which we can call "child entropy".

4. _best_split Method: It finds the best feature to split the data and the best threshold value to split the data by considering all possible splits on all features. It will loop through each feature and its unique values as a potential threshold and then it will calculate information gain associated with every split and choose the one that has the highest information gain.

5. _build_tree Method: Recursively builds the decision tree by splitting the dataset depending on the best splits that were found. Verifies stopping conditions: maximum depth was reached, a node is pure-all samples have the same class, and too few samples to effectively split. In the case of not meeting any of those stopping conditions, the

function splits the data into subsets, instantiates left and right child nodes, and runs the function recursively.

6. _most_common_label Method: This determines the most frequent class label among the samples in y. This is used internally to set a value for a leaf node, when further splitting either isn't possible or wouldn't be beneficial. So this will ensure that a leaf node will predict the majority class of its samples.

7. fit Method: The function trains the decision tree classifier using the training data (X and y). It starts recursion to build the tree by calling the method _build_tree starting from the root node with initial depth equal to zero. Such a tree structure's result is then stored in self.root.

8. _traverse_tree Method: This function navigates the decision tree in order to make a prediction for a single sample, x. Moving from the root node down through left or right child depending on whether the sample's feature value is less than or equal to the node's threshold. It continues this process until it reaches a leaf node and then returns the class value at that leaf node as its prediction.

9. predict Method: Generates predictions for multiple samples. It applies _traverse_tree to each sample in input array X, collecting the predicted class labels into an array that it returns as output.

10. accuracy Method: Evaluates a decision tree classifier against some dataset, y, by comparing predicted class labels against the true labels, y, and calculates the proportion of correct. It gives an overall measure of how well this model is performing.