README FILE

Shirsha Chowdhury

18EE10068

Imports Used:

Pandas to read the dataset

Math for mathematical tasks used for formulas

Functions used:

Entropy(p): Calculates and Returns the value of Entropy when probability=p

sumArr(arr): Returns the sum of a array

sumEntropy(d,N): Calutates the total weighted entropy. d is a dictionary which contains the different classes of a attribute as a key. Key holds a array which gives the count of different target classes of that particular particular attribute class.

totalEntropy(dataset): Calculates the total Entropy of the dataset.

bestSplit(dataset): Returns the The attribute at which best split occurs according to Information Gain. This function first calculates the Information Gain of all the attributes of the dataset and then returns the attribute with information gain.

base1(df): First Base Case is to check if all the target values of a dataset belong to the same class. If that happens we terminate by creating the leaf node with value as that of the target class

base2(df): Second Base Case is used when all the attributes have been used and the value of leaf node will be determined by the majority class present in the target column

trainTree(df): This function Builds the tree recursively. First base1 and base2 function is called to check if a termination case is encountered, in that case the branch is terminated with the class value. Else bestSplit function is called to get the attribute with highest Information Gain. After that new datasets are created by splitting the tree according to the best attribute. The new dataset is again passed onto the trainTree function and the retuned tree will be the subtree of the original tree.

predictRow(arr,tree): for each training example(i.e each row) of the training set it returns the ouput class for that training example.

predict(test): This function loops over the training dataset and returns a array containing the class value of all the training exmaples.

accuracy(res,test_target): Predicts the accuracy of the training set by the formula=$\frac{\text{Total Correctly Matched}}{\text{Size of The Training Example}}$

printTree(d,tree): Prints the tree in a readable format. Initial depth (d) is equal to 0. Eg of output:

```
---------maint=med
------------lug_boot=small
--------------doors=2:unacc
--------------doors=3:good
--------------doors=4:good
--------------doors=6:good
------------lug_boot=med
--------------doors=2:good
--------------doors=3:vgood
--------------doors=4:vgood
--------------doors=6:vgood
------------lug_boot=big:vgood
```

Tree Structure:

The tree is stored in dictionary format with key storing the branch question for eg: lug_boot=small or maint=high . The subsequent key value can be a string stating the class if the tree terminates or it can be another dictionary which is the subtree of the original tree. For eg:

```
"maint=med": {
    "lug_boot=small": "unacc",
     "lug_boot=med": {
         "doors=2": "unacc",
         "doors=3": "acc",
         "doors=4": "acc",
         "doors=6": "acc"
      },
     "lug_boot=big": "acc"
  }
```

Note:

In the test set some values of doors were equal to 56, that was changed to 6.
