Machine Learning: Decision Tree PA

Learning Objectives

After completing this activity, students should be able to:

- Construct an object that can build a decision tree
- describe how tree depth relates to under and overfitting.

In this assignment, you will build a Python object to build a decision tree that will perform binary classification. The name of this object must be *BinaryDT*.

The minimum methods your decision tree class needs to support follows:

__init__	**X**	An m by n numpy matrix that is organized by examples (rows) and features (columns). Data will be preprocessed so that X contains only **numeric** values. That is, binary attributes will be changed to 0s and 1s and ordinal attributes will be changed to enumerated to [0, 1, 2, ...].
	y	A length n numpy array that contains the class labels for each corresponding row in X. (0 = false, 1 = true).
	maxDepth	The maximum depth of the decision tree
predict	x	Accepts a length n numpy array and, using the generated decision tree, returns a tuple with the first entry being the predicted class label (0 or 1) and the second entry being the *entropy* of the leaf node that was reached.
print		Print your decision tree. I will use this data for testing, so, it must match the format *exactly.*

Formatting the Tree Description

Your print method must print a text description of the resulting decision tree formatted, with one line per node, exactly as follows.
A header line that reads:

nodeId,depth,leftNodeId,rightNodeId,splitDim,splitPoint,entropy,infoGain,class0,class1,totalSamples

where:

nodeId	A nonnegative unique integer used to ID this node in the tree
depth	Depth of the node in the tree [0..]
leftNodeId	the nodeID of the left child. If this is a leaf node, assign -1
rightNodeId	the ID of the right child. If this is a leaf node, assign -1
splitDim	a value from between 0 and $d-1$ that shows which dimension was used for this split
splitPoint	The value that is compared to <=, and if true, the tree is descended to the left
entropy	the entropy of this node rounded to 3 decimal places
infoGain	the information gain from this split (parent - sum of children). If this is a leaf, assign "-1.0"
class0	number of examples of class0 in this node
class1	number of examples of class1 in this node
totalSamples	sum of class0 and class1

Collaboration Policy and Package Usage

For this project, you are only allow to use the following python packages:

- numpy
- math
- sys
- pandas

This is an **individual** assignment, and while helping your colleague to find a bug or to help them understand some concepts is OK, sharing of code is prohibited. You should be able to explain all of your code to me, as failure to do so will be an indicating that you over-collaborated. This is machine learning class, so, do not be surprised that I am using text mining tools to detect similar code to the class and to Internet sites.

If you have questions about this policy, please see me.

Submission

You will be submitting your assignment to Autolab (`autolab.cs.jmu.edu`). To get your password, just use the *Forgot my password* link on the home page. **You can only access Autolab from on campus or by using the JMU provided VPN**. A link on the resource page for this class illustrates how to connect to Autolab via the VPN.

Autolab will assess your program using the sample datasets and some unpublished test sets.

Sample Output

Four sample files are available for download from the assignment in **canvas**. They are a modified version of the wine dataset which has been reduced to only having 2 classes (0 and 1). Pairs of files are for training (data and labels) and testing (data and labels). The output from my baseline program running print() after building a tree of depth 2 are in a file named *winePrintOutput.csv.* Running your tree building method and print() must yield exactly these results *EXCEPT* the node numbers need not match (but they need to make a tree, e.g., nodeRightID should have the nodeId of some row in your output).

Using this built tree, I then ran predict() on each of the data points in the test dataset. A file named winePredictTestOutput.txt is provided for reference. It shows the actual label, y, along with the tuple that is returned ($yhat$, and the entropy of the leaf node). I also noted the accuracy of the tree on the test data (which was $\approx$ 96%). Recall that the format of this file is not specified, as I will be calling your *predict* method and evaluating the returned tuple to see if it is correct (no printing required).

Rubrics

The following is the breakdown how this assignment will be graded:

Item	Description	Points
Readability/Style	Program is commented, following naming style for either Python or Java. Points will be deducted for poor variable names or unreadable code	10
Poor Performance	Correctness and clarity for this assignment is much more important than efficiency. However, your program should be able to build the decision tree within 20 seconds of CPU time.	10
Correct Output for published test sets	Programs output matches published output. Small mismatches in output will result in a 5 point deduction for each occurrence.	45
Correct Output for 3 unpublished Test Sets	Programs output matches . Small mismatches in output will result in a 5 point deduction for each occurrence.	15
Correct Output for predict() function	Function returns the correct value for samples from unpublished data sets	15