Big Data Analytics Assignment 2

Implement and train a decision tree

**Task:**

Using the data matrix that you have created, you will now create a simple decision tree model to learn how to predict the labels you have given your data. The long term goal is to create a MapReduce random forest implementation, but the first step in accomplishing that goal is to make a simple decision tree weak learner. This weak learner provides the foundation for the later more advanced ensemble method.

This decision tree does not need to do sub-tree raising or pruning – it is a simple and direct decision tree that maximizes information gain at every split.

**Minimum deliverables:**

Source code that creates the decision tree

README.txt if more than one source file

Summary statistics that shows the performance of your decision tree on a sample of your data

**Due Date:**

One week from assignment

**Requirements:**

The decision tree should be implemented in Java, so that it can later fit into the MapReduce paradigm natively. It is possible to implement your tree in other languages and wrap them in Java, however that is not recommended as it potentially will create runtime problems down the road. Your code must be commented, and if there is more than one source code file, you must include a README.txt file that explains the relationship of all source code files.

You must split your data into a training and testing set for verifying your decision tree functionality. The split must be 80% train, 20% test. The sample should be no less than 10,000 instances, but can be larger.

**Hints**:

* Stick with a binary tree – you should not do more than one split at any node of the tree
* Keep your data structures clean! Adhering to good object oriented principles and taking a little extra effort to comment your code well can go a long way
* Don’t forget that you test ALL the features for information gain at each split
* There are plenty of tutorials for classification information gain readily available. A little different is growing the regression tree.  
  The basic algorithm for growing a regression tree is :
  + Start with a single node containing all points. Calculate mc and S.
  + If all the points in the node have the same label then stop. Else search all binary splits of all features for the split that will minimize the sum of squared error at that node as much as possible.
  + Use the split to create two new nodes.
  + Iterate this process for each node until all nodes are pure or as pure as possible.

**Readings and Resources:**

*Probability*

http://www.autonlab.org/tutorials/prob.html

http://www.autonlab.org/tutorials/pdf.html

*Information Theory*

https://www.khanacademy.org/computing/computer-science/informationtheory/info-theory/v/intro-information-theory

https://www.cs.cmu.edu/~roni/10601-slides/info-theory-x4.pdf

http://motherboard.vice.com/read/how-digital-information-unifies-quantum-mechanics

*Information Gain*

http://www.autonlab.org/tutorials/infogain.html

*Decision Trees*

http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-20/www/mlbook/ch3.pdf

http://www.autonlab.org/tutorials/dtree.html