CS4361/5361 Machine Learning

Fall 2019

Lab 2 - Decision and regression trees

Due Monday, September 23, 2019. Submit a paper copy by 5:00 p.m.

Email report to *olacfuentes@gmail.com*, include UTEP-ML2019 in the subject line.

Your task for this lab is to implement decision and regression trees. The file *magic04.txt* contains data from a gamma ray detection experiment. Each line in the file represents an observation, with the first ten items, all floating point numbers, describing the data collected by the detector, and the last item, a character (g or h), indicating whether the detection corresponds to a gamma ray (g) or not (h).

The program *decision_tree.py* provides code that reads the data, splits it into training and testing sets, and implements a one-node decision tree from the training data to classify the test data.

For this lab, you will extend the decision tree functions as follows:

1. Extend the *_build_tree* function. In the current implementation, *left* and *right* are classification labels (0 or 1); in the extended implementation, they should be references to decision trees, provided the dataset is large enough and the goal accuracy has not been attained.

2. Extend the DecisionTreeClassifier class to consider multiple thresholds for each attribute instead of just the mean. To do this, you can generate several threshold values at random in the interval $[min(x[:, i]), max(x[:, i])$ for each attribute i.

3. (Extra credit for CS4361 students, mandatory for CS5361). In the current implementation, we only consider one threshold for each attribute, the average (or mean) value of that attribute over the training set. Modify the *_build_tree* function to choose the attribute and threshold combination that yields the highest information gain, as described in Mitchell's book. This selection can be time-consuming, thus you should try to make it as efficient as possible. The optimal threshold selection for a given attribute can be done in $O(nlogn)$, while naive implementations take $O(n^2)$; also try to avoid loops if possible.

4. Based on the DecisionTreeClassifier class, implement a RegressionTree class that performs regression instead of classification and test it using the solar dataset from lab 1. A leaf in a regression tree is the average target function value of the examples associated to that leaf during training. To choose the attribute and threshold associated with each internal node, consider the mean value of the attribute as threshold and choose the attribute that results in the smallest mean-squared error.

5. Extend the RegressionTree class to consider multiple thresholds for each attribute instead of just the mean. One possibility is to consider each attribute value as a possible threshold, another is to randomly generate several candidate thresholds.

Write a report including (at least) the following items:

1. Problem description
2. Algorithms implemented
3. Experimental results, including accuracies or mean squared errors and running times for each algorithm and parameter choice. You may want to use tables and or plots to illustrate this.
4. Discussion of results. How do trees compare to k-nn in terms of accuracy/MSE and running times? What parameter choices work best? What parameter choices result in overfitting?
5. Conclusions
6. Appendix: Source code