COMS 4771: Machine Learning | Due: Feb 23^{th} 2016

Homework 2 - Solutions

Instructor: *Satyen Kale*

Part 1

Decision Tree training is a top-down and greedy approach.
top-down: It begins at the top of the tree (at which point all observations belong to a single region) and then successively splits the predictor space; each split is indicated via two new branches further down on the tree.
greedy: At each step of the tree-building process, the best split is made at that particular step, rather than looking ahead and picking a split that will lead to a better tree in some future step.
Algorithm:

1. Select the best feature $(x_1, x_2, x_3, \ldots)$ and threshold to split. We choose the feature and threshold that has the maximum reduction in uncertainty. When there are two features x_i and x_j that both achieve the maximum reduction in uncertainty, we choose the lower indexed one as the feature to split on. In this assignment, we only choose thresholds among the values of the feature in the training set. There are three notions of uncertainty: classification error, gini index and entropy.

   In the assignment, the label is binary: $y \in \{spam, not\ spam\}$.
   If in a set of examples S, a p fraction are labeled as spam, then the various notions of uncertainty are computed as follows:

   - Classification error:
   $$u(S) = min\{p,\ 1-p\}$$
   - Gini index:
   $$u(S) = 2p(1-p)$$
   - Entropy:
   $$u(S) = p \log \frac{1}{p} + (1-p) \log \frac{1}{1-p}$$

   Define $\{x \mid x_j \leq t\}$ as the region of predictor space S in which the feature x_j takes on a value less than or equal to t.
   For any j and t, we define the pair of half-planes
   $$S_L(j,t) = \{x \mid x_j \leq t\} \qquad \text{and} \qquad S_R(j,t) = \{x \mid x_j > t\}$$

and we seek for the value of j and t that maximize

$$u(S) - (p_L \cdot u(S_L) + p_R \cdot u(S_R)),$$

where $p_L = |S_L|/|S|$ and $p_R = |S_R|/|S|$.

2. Stop if a stopping criterion is reached (in this assigment, the stopping criterion is depth of the decision tree) or all labels in leaf node are the same.

3. Assign leaf with majority vote on training examples in leaf.

4. Go to 1. if not yet stopped.

5. Once the regions $(S_1, S_2, \ldots, S_J)$ have been created, we predict the response for a given test observation using the label of the leaf node it belongs to.

Results of computing training and test error with the uncertainty measure of classification error on the `spam_data.mat` dataset:

	depth = 1	depth = 3	depth = 6
Number of training errors (Training Error Rate)	729 (0.2024)	360 (0.1000)	252 (0.0700)
Number of test errors (Test Error Rate)	217 (0.2170)	120 (0.1200)	97 (0.0970)

Results of computing training and test error with the uncertainty measure of gini index on the `spam_data.mat` dataset:

	depth = 1	depth = 3	depth = 6
Number of training errors (Training Error Rate)	730 (0.2027)	367 (0.1019)	227 (0.0630)
Number of test error (Test Error Rate)	219 (0.2190)	126 (0.1260)	102 (0.1020)

Results of computing training and test error with the uncertainty measure of entropy on the `spam_data.mat` dataset:

	depth = 1	depth = 3	depth = 6
Number of training errors (Training Error Rate)	730 (0.2027)	456 (0.1266)	251 (0.0697)
Number of test errors (Test Error Rate)	219 (0.2190)	142 (0.1420)	87 (0.0870)

Part 2

Results of computing training and test error with different passes on the `spam_data.mat` dataset, using the convention that if the dot product is zero, then the prediction is -1:

	passes = 1	passes = 10	passes = 50	passes = 100
Training error rate	0.4771	0.3541	0.2888	0.2610
Test error rate	0.4450	0.3380	0.2790	0.2490

In the following pseudocode, we use the notation $[x, 1]$ to denote the vector x augmented with an extra coordinate with value 1.

Algorithm 1 Online Perceptron (training and testing)

```
Training
Let d be the dimension of the feature vectors
initialize w = 0⃗ ∈ ℝ^{d+1}
for each pass in passes do
   for each training example (x_i, y_i) do
      if sign(w · [x_i, 1]) ≠ y_i then
         w = w + y_i[x_i, 1]
      end if
   end for
end for
return w
Testing
Set loss = 0
for each test example (x_i, y_i) do
   if sign(w · [x_i, 1]) ≠ y_i then
      loss = loss + 1
   end if
end for
return loss
```