# B551 Fall 2012 Homework 4

### Due date: 10/25/2012

The problems below will ask you to implement three machine learning strategies for a text classification problem on a Reuters news dataset. (See readme.txt for more information about the framework code).

Given a set of positive/negative topics, the programming framework will automatically extract training/testing examples from documents in the dataset. The text documents are processed into fixed-length examples using *bag-of-words* features. This approach takes a document $d$, a list of $F$ words $w_1$, …, $w_F$ and outputs the example x=($f_1$, …, $f_F$) where $f_i$=1 if $w_i$ appears one or more times in $d$, and 0 otherwise.

## 1) Naïve Bayes Classifier (30 points)

You have already seen Naïve Bayes classifiers as part of more general Bayesian networks in HW3, and now you will implement one from scratch.

A. Implement the Naive Bayes classifier in the NaiveBayesClassifier class in classifiers.py by implementing the train() and test_one() functions. For training, use a uniform class prior (rather than performing maximum-likelihood on the class counts). Learn the MAP estimates of each feature probability given a Beta prior with hyperparameters a = b = 2 (i.e., 1 virtual count for both positive and negative examples).

   Your test_one() function should work correctly even for nonuniform class priors and different parameters for the Beta feature priors.

B. Evaluate your classifier, where the positive examples are given by the 'earn' topic, and the negative examples are given by the 'acq,crude,gold' topics.

   Use the --trmax=N command line option to vary the number N of examples in the training set by increments of 100 from N=100 to 1000. Plot the accuracy on both the training set and test set against N. Include this plot in your answer document (it should contain two learning curves, one for training set accuracy and the other for testing set accuracy). What do you observe?

C. Repeat the process of question B, but use the --fsize=F option to classify.py to vary the number F of features (words) extracted from the documents with F=5, 10, 20, and 50. Plot the accuracy on both the training set and test set against F as you did before. What do you observe?

## 2) Decision Tree Classifier (40 points)

A. Implement the decision tree learning algorithm in the DecisionTreeClassifier class in classifiers.py, by implementing the decision_tree_learning() and choose_feature() functions (train() and test_one() are already given). To pick attributes to split on, you may use either the

minimum error criterion given in class, or the information theoretic criterion given in R&N p. 703-704. Do not perform pruning.

B. Same questions as 1.B. using this classifier.

C. Same questions as 1.C. using this classifier.

D. (Written) Suppose we generate a training set D that is consistent with a decision tree *T* (say, by randomly sampling branches). If we apply decision-tree learning to *D*, is it the case that the learning algorithm will eventually learn a tree *T'* that is consistent with *D* as the training-set size goes to infinity? Is it the case that the learning algorithm will eventually return the original tree *T'* = *T* as the training-set size goes to infinity? Why or why not?

E. (Written) Consider the following data set comprised of three binary input attributes $A_1$, $A_2$, and $A_3$ and one binary output:

| **Example** | **$A_1$** | **$A_2$** | **$A_3$** | **Output y** |
|---|---|---|---|---|
| $x_1$ | 1 | 0 | 0 | 0 |
| $x_2$ | 1 | 0 | 1 | 0 |
| $x_3$ | 0 | 1 | 0 | 0 |
| $x_4$ | 1 | 1 | 1 | 1 |
| $x_5$ | 1 | 1 | 0 | 1 |

Trace by hand the steps taken by the DTL algorithm on R&N p.702 to learn a decision tree for these data. List the major steps and show the computations made to determine the attribute to split at each node.

## 3) Custom Classifier (30, possible 10 points extra credit)

A. Implement a third classifier. Implement your learner ThirdClassifier in classifier.py, and you may optionally implement any new feature extraction techniques in FeatureExtractor() in utils.py. Describe your new technique and your rationale for choosing it.

Suggestions for possible techniques include decision tree pruning, boosting with decision stumps, Bayesian networks. Or, you could tune your existing learners. Another option is to select features that better discriminate between topics (e.g., ignore common words like "the", "is", or "of").

Extra credit will be awarded for the classifier with the best results on a new classification task. Note that you will not be tested on the same categories as in questions 1) and 2), so make sure your techniques work for different categories. We will also award EC for creative implementations.

B. Same questions as 1.B. using this classifier.

C. Same questions as 1.C. using this classifier. (If, for some reason, you have used a new feature selection technique that does not permit variable numbers of features, explain why this is so, and skip this question.)

## Submission Instructions

Save your plots to a PDF document hw4_answers.pdf. Upload your hw4_answers.pdf, classifiers.py, and any other Python files you changed to OnCourse.