EECS 349 HW 5

Marc Gyongyosi

Problem 1:

The naïve Bayesian Classifier is implemented in the included python file.

Problem 2:

The code is included in the file.

Problem 3:

Proof that (6) and (7) are equivalent if no overflow:

Given:

$$\underset{v_j \in V}{argmax}(v_j) = \underset{v_j \in V}{argmax}(\log(v_j))$$
$$\log(A * B) = \log(A) + \log(B)$$

(6) and (7) are then equivalent because you can split up multiplication into log addition, and because the argmax will still remain the same between the two compare ones (although the numerical values of the probabilities will obviously not be the same).

The problem with underflow precision is that as we multiply the very small probabilities, the probability measures get too small to represent. On a 64 bit machine, the smallest denormalized number that can be represented as a double is 2^-1074. As we multiply potentially thousands of probabilities, the loss of precision will negatively impact the performance of the algorithm – i.e. if any of these probabilities becomes 0 because of rounding the entire calculation will be off and effectively go to 0. That's where the logarithmic addition comes into play. This effectively eliminates this negative effect of one of the probabilities becoming 0 due to rounding because the log of a number in]0, 1] is a large number.

Problem 4:

I set up an experiment which can be run from the spamfilter_exp.py file.

a) **Dataset**: I use the two files that were previously used and also suggested in the assignment to test the performance of the spam filter. This program does n-fold cross-validation, without having to copy around any of the files. I do this, by creating and modifying my functions so that they can handle input only consisting of filepaths, instead of directories. The ratio of spam vs ham is roughly 1:4 (~500:~2000). I decided this data set because it was indicated in the assignment that the filter would work well with this data. Looking through a few of the files, I believe that they seem to be very realistic in the way the spam looks. However, I think that one caveat is the low percentage of spam – in the real world this is probably much much higher.

b) **Methods**: I test by creating a dictionary with the n-1 testing groups and then running the spam predictor on the remaining testing group. This was done n times. (I run all tests with 10 fold cross validation, however, the program supports arbitrary numbers for this and will

construct the testing and training sets accordingly). To compare against prior probability, I compared how often (with the entire data) the spam predictor was wrong (i.e. missed a spam file or misclassified a ham file).

c) **Results:**
this is the output of the experiment program:
using 10 fold cross validation with one group containing 305 items
using 50 samples for spam and 255 for ham

Run: 0
Correctly identified as spam: 50 out of 501
Wrongly identified as spam: 4 out of 2551
Error rate: 0.149082568807
Prior Probability Error: 0.164154652687

Run: 1
Correctly identified as spam: 50 out of 501
Wrongly identified as spam: 11 out of 2551
Error rate: 0.151376146789
Prior Probability Error: 0.164154652687

Run: 2
Correctly identified as spam: 50 out of 501
Wrongly identified as spam: 15 out of 2551
Error rate: 0.152686762779
Prior Probability Error: 0.164154652687

Run: 3
Correctly identified as spam: 50 out of 501
Wrongly identified as spam: 13 out of 2551
Error rate: 0.152031454784
Prior Probability Error: 0.164154652687

Run: 4
Correctly identified as spam: 49 out of 501
Wrongly identified as spam: 3 out of 2551
Error rate: 0.149082568807
Prior Probability Error: 0.164154652687

Run: 5
Correctly identified as spam: 50 out of 501
Wrongly identified as spam: 10 out of 2551
Error rate: 0.151048492792
Prior Probability Error: 0.164154652687

Run: 6
Correctly identified as spam: 50 out of 501
Wrongly identified as spam: 10 out of 2551
Error rate: 0.151048492792

Prior Probability Error: 0.164154652687

Run: 7
Correctly identified as spam: 50 out of 501
Wrongly identified as spam: 14 out of 2551
Error rate: 0.152359108781
Prior Probability Error: 0.164154652687

Run: 8
Correctly identified as spam: 50 out of 501
Wrongly identified as spam: 12 out of 2551
Error rate: 0.151703800786
Prior Probability Error: 0.164154652687

Run: 9
Correctly identified as spam: 50 out of 501
Wrongly identified as spam: 8 out of 2551
Error rate: 0.150393184797
Prior Probability Error: 0.164154652687

As you can see, the experimental results indicate that the naïve classifier is only slightly better than prior probability. The error rate on the Bayes classifier is about 1% better than prior probability. I expect this to fluctuate with the dataset used, thus, I don't think this system is scalable.