Project 5

ReadMe

Mike Roylance
roylance@uw.edu

<table>
<tr><td>Language coded</td><td>Java</td></tr>
<tr><td>Approach</td><td>I created 3 classes to help me complete this project
<table>
<tr><td>App.java</td><td>This class was in charge of reading information from the file system. It also printed out the information to the console.</td></tr>
<tr><td>BayesianReporter.java</td><td>This class was in charge of counting up the probabilities for each sentence. It takes in all the languages (in a nested HashMap) and introduces LaPlace smoothing for calculating the probability of a word belonging to a particular language.</td></tr>
<tr><td>Tests.java</td><td>This class tests the logic in Bayesian Report.</td></tr>
</table>
</td></tr>
<tr><td>Smoothing</td><td>I decided to use LaPlace Smoothing for this problem. I find that metho effective and easy to implement:

$$\hat{\theta}_i = \frac{x_i + \alpha}{N + \alpha d} \qquad (i = 1, \ldots, d),$$

On line 20 of BayesianReporter.java, I am adding in the denominator portion (1 for every word).

On lines 54 and 62, I'm adding in the numerator addition (+1). This ensures that I will never have a probability of 0.</td></tr>
<tr><td>Extra Credit</td><td>I created a function CalculateMaximumUnknownThreshold that sets a threshold of "all but 3" of the words being unknown. In other words, if only three words are known, it's too little and the sentence is classified as "Unknown". This is calculated by the LaPlace smoothing of (1 / TotalCount) where TotalCount is n + k (n is the total domain count where k is the unique count). This is stored in the extra-credit.txt file.</td></tr>
<tr><td>Results</td><td>Please look at output.txt created from condor.cmd for more information</td></tr>
</table>