Author: Ryan Timbrook
UW Net ID: timbrr
Project: Ling 473 Project 5
Date: September 8, 2016

Description:

This project was setup to build a naïve Bayesian classifier that would classify text fragments, sentences, according to their language. The total sample space was composed of 15 unique language sub-sample sets which included each languages 1500 most commonly occurring words. The subsample word counts varied by language which presented a challenge when smoothing for non-found text from the input data.

Approach:

To accomplish the various data scoping challenges this task presented I took a more dynamic programming approach than I have in prior projects. For this effort I created two custom classes in a separate python module then from the core processing script. These classes act as memory value objects along with having utility functions for processing attribute information.
The 'LMFrequencyDist' class is used to store and perform utility functions on the language model sample data. For each language model a unique instance of this class is created at runtime as a preprocessing step. The 'TextProbDist' class handles similar value object memory storying as the 'LMFrequencyDist' object but also includes probability functions used by the main script to identify the language label for a given text fragment. It also has a printOutput function which outputs the required data for this projects submission. The main processing along with preprocessing and cleaning of the input data of punctuations is handled by the python script 'ling_473_pr5.py'.

Challenges:

Because of the large data set size the probability functions were challenging to setup. After various attempts with log10 combined with tweaks to the smoothing function, what I ended up with gives the correct identified language based on the train.txt file provided for validation.
My approach to smoothing was to identify any non-word match as a singleton over the sample space. This included having 1 as the numerator of the probability equation and the language sub-sample space word occurrence count as the denominator. However all of my early attempts with this smoothing approached, both intuitively and result output wise, did not work and through off the entire probability space. The issue was that I was using the language specific sub-sample space's word occurrence count totals. These totals are not uniform which caused smoothing to not work. Taking the entire sample spaces word occurrence counts, the sum of all sub-sample spaces word occurrence counts, and using that value as the denominator fixed the smoothing issue so all non-found words had an equally likely probability over all languages.