**General design of System**

This is an improved spam filter model based on Naïve-Bayes model, in which every feature is independent of each other given its category. My system involves the following optimizations and processes:

1. Preprocessed tokenized method
2. Unigram feature categorization
3. Bigram feature categorization
4. Length feature detection
5. Experimentation on different smoothing constants

The error rate is 1% on developed spam set, and 0.5% on developed ham set, for a total average of 99.25% accuracy.

**Experimented Features**

1. Unigram and Bigram features
2. Length of the words features (Separating length to different entries within the dictionary)
3. Different smoothing constants for unigrams vs. bigrams.

**Pre-processing and tokenization.**

Pre-processing and tokenization took place in the log\_probs function, which aimed to categorize the training data in a single pass based on unigram/bigram possibilities and length of unigram words.

The email contents were separated by the means of delimiting white-spaces, then for each word, we increment count based on the length of the unigram word (less than 8 characters, 8 to 19 characters, and at least 20 characters). We also separately record the frequencies of the unigram words themselves. In addition, for all words other than the last character in a particular email file, we also record the frequency of the bigram consisting of that word and the word immediately following it.

Finally, following the Bayes paradigm of 

We record the logarithmic result into the dictionary, including those specifically related to word length as well as bigram/unigrams.

**Experiments and Results**

I experimented with the following ideas: smoothing constant, frequency of occurrence for words, and here are the experimental results:

Initially, I changed smoothing uniformly from 1e-5 to 1e-100, and it resulted in the following changes:

94.5 -> 96.5 in train/spam

99.5 -> 97.5 in train/ham

Average: 97% -> 97% no change

Next, I experimented with changing uniform smoothing from 1e-5 to 1e-12:

94.5->95, 99.5-> 99.5

average 97.25 slight improvement

Finally, I experimented with changing smoothing for unigram to 1e-15, and bigram to 1e-19

Also resulted in only slight improvements

The results demonstrated that while smoothing constants provide marginal improvements, such implementation cannot directly cause significant improvements in the overall system design that adheres to the Bayesian model.

In addition to the results in the model, I also experimented with separating features based on digits, such that words with only digits are stored in addition, similar to the words length:

Interestingly, the modification resulted in the following changes in the spam/ham datasets, from 94.5, 99.5 to 95.5, 88 which was a worse modification. I suspect that this is also due to a mishandling of code implementation in the beginning of the project, however.

**Analysis of the errors made by system on the development data.**

**Spam Dev Example: dev222**

This spam file is very short, which indicates that there is less inference and collection of probabilities to be made from the Bayesian model. Moreover, this spam file is filled with an even mix of differently length words, which makes it even harder to detect as a spam, given the short length of the email. Finally, one of its words is a very long url link, which would be an anomaly, except the occurrence of a single word longer than 20 characters would be difficult to notice without a more sophisticated classifier system.

**Ham Dev Example: dev118**

This file in particular has a lot of words with length less than 8. The relative frequency is much more than those of other training sets on non-spam emails, which causes the is\_spam method to falsely detect such behavior as an anomaly to an extent such that it compensates for all other factors considered by the model.