**MULTINOMIAL**

I decided to use what the lecture slides called the "somewhat more subtle version" of smoothing to prevent over-fitting, where I used an alpha instead of one. My first chosen alpha was 0.5. I also used log probabilities instead of just probabilities to prevent underflow.

My first test run was an accuracy test. I wrote a function to pick the best category for a message feature based on my Multinomial Classifier, then ran it on twenty message features per category. In a test set of 20 categories, this meant I scored 400 features. 373 features were classified correctly, for a total of 93.25% correct classifications.

I then attempted to test different alphas to see how I could improve my accuracy, with these results:

|  |  |
| --- | --- |
| Alpha | Accuracy |
| 0.7 | 92.75% |
| 0.6 | 93.00% |
| 0.5 | 93.25% |
| 0.4 | 93.50% |
| 0.3 | 94.25% |

Unsurprisingly, accuracy improves as I get smaller alphas (and thus closer to over-fitting). I will leave my alpha at 0.5 for now and adjust the alpha when I reach the k-fold section.

\*Note on the multinomial code: I used an old CS124 homework assignment as a guide.

K-FOLD

I started the k-fold on the multinomial classifier, without feature selection, and with alpha = 0.5. I got an average accuracy of 83%. I then decided to test new alphas to try to improve this accuracy, yielding this graph:

This (roughly quadratic) trend was the same as the prior trend; the lower the alpha, the higher the accuracy. However, too small is over-fitted and 1 is, according to lecture slides, simplistic; so I will continue to use my average of 0.5 as an alpha.

Error for multinomial classifier without feature selection and alpha 0.5: 17%