COMP 472 Spam Filter Project Final Report - Deliverable 3

Daniel Miller - 6602002
Yash Lalwani - 6531857

We certify that this submission is the original work of members of the group and meets the Faculty's Expectations of Originality.

This report describes the Spam Filter project and discusses the effect of the changes made, with respect to the original decisions.

When choosing what constituted a ‘word’ in our original program, the following were the most important assumptions we used:

1) Words must have a minimum character length of 3
2) Words cannot contain digits
3) Hyphenated words are treated as separate words
4) Anything between “<>” is treated as an HTML tag
5) A list of words commonly found in the email header are ignored

After reading all the emails and creating lists of all the words, we run through our lists to determine the ham and spam probabilities of each word, and the smoothing technique we used was **add 0.5.**

Our training set consisted of 1000 ham emails, and 1000 training emails.

We ended up with a model consisting of **23,252 unique words**.

Classifying our test emails gave us the following results:

		Predicted Class	
		Ham	Spam
Actual Class	Ham	394	7
	Spam	35	365

Average accuracy = 759 / 801 = 94.76%

We believe this is a very good level of accuracy, especially considering the fact that our word-detection tests can only take care of so many words used in email headers.

Moreover, for all **Ham emails**, we had a **98.25% accuracy**, and for our **Spam emails** we had a **91.25% accuracy**. Thus, despite having some wrong classifications, since they affected Spam a lot more than Ham, the probability of the user missing an important email is much lower than the probability of reading a spam email. Although this isn't ideal, it is still preferable to the former probability being as high as the latter.

For Deliverable 3, we decided to change one of our assumptions for detecting words, and our smoothing technique. After each change, we rebuilt our model and ran our classifier again. The sizes and contents of our training and test sets remained the same.

1) **Changed minimum word length to 4 characters**

We chose to change the minimum length of a word to 4 characters rather than 3. We thought this was a reasonable change since we could eliminate several common words such as 'too', 'the' and 'for'.

We ended up with a model of **21,741 unique words.**

Classifying our test emails with the new model gave us the following results:

		Predicted Class	
		Ham	Spam
Actual Class	Ham	394	7
	Spam	34	366

Average accuracy = 760 / 801 = 94.88%

We observed that we didn't obtain a large difference from our original results. Our average accuracy increased slightly. The new model did not affect our Ham accuracy at all, but increased our accuracy for predicting Spam emails from **91.25%** to **91.5%.**

We weren't actually surprised, as we had predicted that the smaller/shorter words were so common that they were more or less evenly distributed between Ham and Spam emails, thus making less of an impact in the overall classification while longer words could provide greater information gain. Our new results mainly proved this observation, albeit to a small extent.

Overall, we don't think this change has a large impact on the spam filter. One benefit is that the smaller word count allows the model to be built faster (although this wouldn't be done frequently for any spam filter), from **90 seconds for the original version**, to approximately **60 seconds for the current version**.

2) **Changed smoothing technique from 0.5 to 1**

Note: This change incorporates change 1, which was to increase the minimum length of a word from 3 characters to 4.

We chose to increase the smoothing factor for experimentation purposes. We were interested in seeing it would affect our results. While we were fairly certain there would be some change in our results with an altered smoothing factor, we were unsure whether accuracy would be increased or decreased and wanted to see concrete data to help us understand the effects. Classifying our test emails with the new smoothing technique gave us the following results:

		Predicted Class	
		Ham	Spam
Actual Class	Ham	395	6
	Spam	37	363

Accuracy = 758 / 801 = 94.63%

As with the first change, we did not obtain a significant change in our average accuracy. In hindsight, this makes reasonable sense to us. Smoothing increases the count for each word by a flat amount in order to prevent zero counts. With a dataset as large as this one, and with some words appearing very frequently, we've now realized that adding 0.5 to each word count would not have much further effect. It seems reasonable that decreasing the smoothing

factor, perhaps to 0.1 or some other small fraction would slightly improve accuracy, however based on current results it does not appear that the increase would be very significant.

As noticed, the accuracy for predicting Ham emails correctly increased slightly, from **98.25%** to **98.50%,** at the cost of the accuracy for predicting our Spam emails correctly, which decreased from **91.5%** to **90.75%**.

We realised that if we had decreased our smoothing technique to adding a value lower than 0.5, we would have gotten a higher average accuracy.

One of the things we think would help us in investigating the effects of the above changes more thoroughly, would be to see which emails' classifications changed between the changes. Then we could go through the weighted probabilities for those emails as well as the probabilities of the words within those emails, and that would give us an idea of the spam filter's decision.

Another thing we would like to do would be to find the differences in the overall Ham probabilities and Spam probabilities for each of our emails. While visually skimming over the 'analysis' and 'results' files, I could tell that for the wrongly classified emails, the difference between their Ham and Spam probabilities was almost negligible, showing that it was a close call. We would like to actually analyse these differences and calculate an error range for our classifier.

A potential improvement would be modifying our Tokenizer to search for standard sign offs within emails, such as ‘Regards’, ‘Sincerely yours’, etc. Determining which of these signoffs are typically more personal and attributing them to being common in ham, spam or both may contribute to improving our accuracy further.

References:

[1] Kosseim, Leila. *Artificial Intelligence: Stochastic Methods* [PDF Document]. Retrieved from https://moodle.concordia.ca/moodle/pluginfile.php/1735784/mod_label/intro/472-4-stochastic.pdf

[2] Calders, Toon. *Classification: Naive Bayes Classifier Evaluation* [PDF Document]. Retrieved from http://wwwis.win.tue.nl/~tcalders/teaching/datamining09/slides/DM09-02-Classification.pdf