AI2

Assignment 1

Lazar Popov (s3340473)
Sander Kaatee (s3150798)
Group 19

September 28, 2020

1-3.3

To improve the method for building up the vocabulary we added the function `clean_up_word` to the class of Bayespam. This function removes punctuation, makes everything lowercase and eliminates words less than 4 letters. It also removes strings of gibberish like lines of equal-signs or IP addresses by removing strings containing certain characters.

Section 2 is implemented within the function `compute_probabilities`. An epsilon of 0.3 gave the best results, however we obtained this by trying some higher values and some lower values, where there seemed no improvement by going lower than 0.3. Perhaps an even better epsilon could be found by using some optimization algorithm such as hill-climbing, however we felt that this was beyond the scope of the assignment.

3.3 Desired Output

	Actual Spam	**Actual Regular**
Classified Spam	66	0
Classified Regular	2	26

Spam correctly classified: 97.06%
Regular messages correctly classified: 100.0%

When switching the input arguments, i.e. using the test folder to train and the train folder to test, the results change to:

	Actual Spam	**Actual Regular**
Classified Spam	62	0
Classified Regular	6	26

Spam correctly classified: 91.18%
Regular messages correctly classified: 100.0%

The difference in results when switching the folders might be due to one or the other having more variability, which could lead to a more/less narrow vocabulary in the training phase. As a consequence, less words make a substantial difference when classifying messages in the testing phase, resulting in a deterioration of performance. This hypothesis is supported by the fact that there is a difference between the size of dictionary when switching folders, namely using the train folder to train results in a vocabulary of 7362 words, while using the test folder to train results in a vocabulary of 7072 words.

4 Naive Bayes classification based on bigrams

4.1

In `bigram_bayesspam.py` we create the bigrams by adding two words from index `i` and index `i+1` together in a token. Before adding those words together we call on the words on the two indexes the function `clean_up_word`. There we adjust the number of letter that is allowed in a word that can be a part of a bigram.

4.2

When we run the file with folders `test` and `train` as parameters the results the result is: Spam correctly classified: 97.06%
Regular messages correctly classified: 96.15%
Which is worse than using no bigrams When we run the file with the folders reversed we get Spam correctly classified: 97.06%
Regular messages correctly classified: 100.0%
This is better than without using bigrams.
Adjusting the parameters
As epsilon became smaller (0.3) the performance increased and it capped at epsilon being 0.3. As epsilon increased the performance decreased. When the minimal word size was set to 4 we achieved 100% correct classification on both spam and regular messages with the folders `train` and `test`. When we reversed the folders we did not get any better than the previous results (97.06% correctly classified spam, 100% correctly classified regular mail). When we increased the size of the dictionary there was no effect when the folders as parameters were `train` and `test`. There was effect when the folders were `test` and `train`. The reason for this was since we train on a data set with less word variety or text richness and we test on a set with bigger word variety or text richness there will be words that have not been seen before. By increasing the size of the vocab and not decreasing it we are trying to get richest vocabulary we can as we are going to be test on a rich vocabulary.

The use of decreasing the size of a vocabulary that is sorted by the frequency of the words that appear is that it decreases the amount of resources needed - big vocabulary would mean more storage. Decreasing the vocabulary is also viable option as the words or even bigram would be distributed in a Pareto manner. So removing words from the tail would remove words that are very rare and that would not affect performance dramatically.

5 Final Questions

5.1 Answers by Sander Kaatee (s3150798)

1. Assuming that there are really no common words (so also no accidental duplicate gibberish), then the classifying will purely be done based on $\log P(\text{regular})$ vs $\log P(\text{spam})$, as there are no

probabilities for the words. As there are more spam messages than regular messages, all messages will be classified as spam.

2. Words are not independent as sentences follow a structure, without which it would not be sentences but instead heaps of gibberish. E.g. after a personal pronoun the odds of finding a verb is far higher than finding a noun, while the odds of finding a second personal pronoun is probably zero. The same goes for the probability of words within bigrams (no bigrams that contain two personal pronouns) and between bigrams (pronoun + verb is more likely to be followed by adjective + noun than by another pronoun + verb).
However accounting for this dependence between words complicates the algorithms a whole lot and increases the computation complexity. This might be useful for harder tasks as sentence generation, but for a simpler task such as a spam classification it is wiser to disregard this dependence.

5.2 Answers by Lazar Popov (s3340473)

1. The e-mail would be classified as spam as there would be no word in the vocabulary. This will make all of the words to be skipped. Since there are more spam messages than regular messages the prior probability of spam would be higher and the message would be classified as spam.

2. The words in a message and in a sentence are not independent. The reason for that is grammar and semantics. There are certain grammar rules that have to be followed in a language which creates a regularity. This will cause words "He is" to appear more often that words such as "He are". The same would be for semantic. Word combinations that make sense would appear more often than words that do not. For example the words "sky blue" would appear more often than "sky green". This shows that the words are not really independent. The same goes for dependence between bigrams and within bigrams independence.