Spring 2022 CS65 Natural Language Processing

Zakir Hossain & Ibrahim Haussouna
Lab 6

zhossai2@swarthmore.edu
ihassou1@swarthmore.edu

April 12, 2022

Answers:

1. We open an email file. Created a message object of the email with the email module. Read each line with the body_line_iterator. Splited the line with whitespace and tokenized the ones that has at least a non-white space character.

2. Tokenized each email from the paths. Aggregated all the tokens from all emails into a single array and counted the frequency of this array with Counter. Iterated over the keys of this frequency dictionary and calculated The the probability for each of the key (which is the unique word). See line 29-41

3. Used the log_probs function to create dictionar of probabilities for spam and ham emails by passing a list of all the paths of the emails contained in a particular directory. Extracted all the paths via the getPath (line) calculated class probabilities for both ham and spam with the formula. See line 54-67

4. Tokenized the given email. Got the frequency of the words in the email. And calculated 2 probabilities, one for ham and the other for spam using the probabilities distribution and the formulas. Which ever one was higher, the corresponding clas was returned. See line 88-104

5. Created a method call most_indicative that takes the N and the

class type. We loop through the types that are common in both probabilities dictionaries and depending on the class type, we used the corresponding probabilities dictionary and class probability and the indicative formula to rank all these common words in terms of the indicative value. At the end we returned the words that have the highest indicative value.

6. About 12 hours.

7. Parsing with the email module was a little tricky. Working in log space without any mistake was a little difficult as well.

8. I liked how accurate the result is, despite being such a simple looking algorithm. P.S: Naive Bayes is not that "Naive" after all.