CSE 537 – Spam Project

Group Member: {Anke Li: 110094670, Yangchun Li: 109819462}

In this project, we implemented Naive Bayes algorithm to classify spam e-mails. We also implemented Laplace Smoothing.

Firstly, we preprocessing the train file in train.py, isolating different kinds of data, and calculate the time of appearance for each word in each type of mails. Specifically, we separate each line and store three types of information into one object of mail: id, type and words appeared in this mail. Then we create three integers and two dictionaries which will be used to identify mails later. Three integers are: number of spam mails, number of ham mails, and number of all mails in train set. Two dictionaries are dict_spam, which contains the number of appearance of each word in all spam mails, and dict_ham, which contains the number of appearance of each word in all ham mails. The actual word is used as key, and the value is the number of times it appeared. Please note that we ignored the time of appearance of one word in one mail, in another word, there are only two possible states for one word in one mail: appeared, or not appeared. The reason is that Naive Bayes algorithm doesn’t take this into account.

Secondly, we analyze the test set in classify.py. We first extracts information from test file, then traverse all mails, calculating the possibility of each mail of being spam or ham. We said one mail is likely to be a spam/ham if the possibility of being a spam/ham is higher than the possibility of being a ham/spam. We also compare our result with the actual type of the mail and count the number of correct and incorrect classifications.

At first, we calculate the possibilities just like the Bayes algorithm. We found that sometimes the multiplying of all possibilities are too small, and finally becomes 0. So we decided to calculate the sum of logarithms of all possibilities. Since logarithm is monotonic, this will not affect the comparing result.

We also get rid of the denominator, since two possibilities have the same denominator.

Result: Our classifier successfully identified 679 mails, so the accuracy is 67.9%.

We also implemented Laplace Smoothing in our code. For example, if one word didn't exist in training spam mails, we set its possibility to be: 1/(count_spam + 2). count_spam is the number of training spam mails.

We found that the result is the same. We also tested just ignoring a word if it didn't exist in one type of mails, and the result still didn't change.

Important optimization:

We have tried two optimization:

1. Take the appearance time of one word in one mail into account. Now the count of one word in spam mails is the total appearance time. For example, if Viagra appeared 3 times in one spam mail and 1 time in other two spam mail, the new count for Viagra would be 5 instead of 3. **If we used the new count to run the algorithm, we could achieve 77.4% accuracy without Laplace smoothing, and 77.8% with it.**
2. Another optimization is to replace the denominator when we calculating the possibility of each word, from the number of spam/ham to the total number of appearance of one word in **ALL** mails. **Using this optimization, we could achieve 86.8% accuracy, with or without Laplace Smoothing.**

The code we submitted includes both two optimizations mentioned above, thus an 86.8% accuracy.