Assignment 5 : Naive Bayes

In this exercise you will be writing a spam detector with Naive Bayes. Download **emails.zip** below. This dataset consists of four sets of data: nonspam-test, spam-test, nonspam-train, and spam-train. You should use the *training emails* to build a Naive Bayes model, and the *testing emails* to test the accuracy of your model. What percentage of emails in nonspam-test does your model predict to be non-spam, and what percentage of emails in spam-test does your model predict to be spam? (these two numbers tell the accuracy of your model).

1. Represent each email by a *set* of unique words (use *set* in python).
2. Exclude [default English stop words]() from each email (If A and B are *python* sets you can exclude members of B from A by subtraction: A-B).
3. Create two dictionaries **nonspam_counts** and **spam_counts.** These two dictionaries keep counts of each word in spam and nonspam emails. Initially for any possible word in the training data do:

   nonspam_counts[word] = alpha
   spam_counts[word] = alpha

where alpha is a number very close to zero (you can initially set alpha to 0.001).

4. For each word in each email of the training data update the counts in **spam_counts** and **nonspam_counts.** (Note that since each email is considered a set, if a word happened a couple of times it is counted only once)
5. Define a function called *classify*. This function takes an email and classify the email as spam or no spam. To avoid dealing with underflow use *logs* and additions in place of multiplication. For example instead of a*b use log(a)+log(b).

Note that
P(spam|email) = P(email|spam)*P(spam)/P(email)
therefore
log(P(spam|email) = log(P(email|spam)) + log(P(spam)) - log(P(email))
similarly
log(P(nonspam|email) = log(P(email|nonspam)) + log(P(nonspam)) - log(P(email))
On the other hand you should classify the email as spam if
P(spam|email) > P(nonspam|email)
or equivalently if
log(P(spam|email) > log(P(nonspam|email)
or equivalently if
log(P(email|spam)) + log(P(spam)) - log(P(email)) > log(P(email|nonspam)) + log(P(nonspam)) - log(P(email))
or equivalently if
log(P(email|spam)) + log(P(spam)) > log(P(email|nonspam)) + log(P(nonspam))

6. Compute the accuracy of the model by calling classify on the test data (what percentage of your calls return the right answer).
7. Now change alpha (make it smaller) , does the accuracy change? try it with a couple of different alphas, what is the best accuracy.

8 (**Optional**). Install and use nltk package for lemmatization and stemming. Lemmatization and stemming should improve your model's accuracy.

Theoretical questions

1. Let *X* be a random variable for coin that comes up heads with probability φ, i.e. $X \sim \text{Bernoulli}(\varphi)$ or $P(X=1) = \varphi$. Furthermore assume there is a prior on φ that follows a Gaussian distribution with mean μ and variance σ, i.e. $\varphi \sim N(\mu, \sigma)$. We flip the coin n times and observe m heads and n-m tails. What is the posterior of X, i.e. $P(\varphi| X_1, X_2, \ldots, X_n)$? Assume that $X_1, X_2, \ldots, X_n$ all follow Bernoulli distributions and are iid.
2. Prove that if $x|y=0 \sim N(\mu_0, \Sigma)$ and $x|y=1 \sim N(\mu_1, \Sigma)$ and $y \sim \text{Bernoulli}(\varphi)$ then $P(y=1|x) = 1/1+e^{\{-w.x\}}$ for a w.