Review Classification using Naive Bayes

Many practical problems involve classifying textual objects—documents, emails, sentences, tweets, etc.—into two specific categories, e.g., spam vs nonspam, important vs unimportant, acceptable vs inappropriate, etc. Naive Bayes classifiers are often used for such problems. They often use a bag-of-words model, which means that each object is represented as just an unordered "bag" of words, with no information about the grammatical structure or order of words in the document. Suppose there are classes A and B. For a given textual object D consisting of words w1, w2, . . . , wn, a Bayesian classifier decides that D belongs to A by computing the "odds" and comparing to a threshold :

$$\frac{P\left(A \mid w_1, w_2, \ldots, w_n\right)}{P\left(B \mid w_1, w_2, \ldots, w_n\right)} > 1$$

where P (A | w1, . . . wn) is the posterior probability that D is in class A. Using the Naive Bayes assumption, the odds ratio can be factored into P (A), P (B), and terms of the form P (wi | A) and P (wi | B). These are the parameters of the Naive Bayes model.

As a specific use case for this assignment, we've given you a dataset of user-generated reviews. User-generated reviews are transforming competition in the hospitality industry, because they are valuable for both the guest and the hotel owner. For the potential guest, it's a valuable resource during the search for an overnight stay. For the hotelier, it's a way to increase visibility and improve customer contact. So it really affects both the business and guest if people fake the reviews and try to either defame a good hotel or promote a bad one. Your task is to classify reviews into faked or legitimate, for 20 hotels in Chicago.
You can run the code like this:

```
[mranka@silo]$ python3 SeekTruth.py deceptive.train.txt deceptive.test.txt
```