Part 2 Explanation:

The way this code works is by going through the training set and calculating the amount of times each word occurs in each classification of spam and ham (not spam). Using these amounts, we were able to calculate the respective probabilities for each word (once again, being classified as either spam or ham).

We then use this information for prediction purposes. For each input, we start with an initial probability of being spam or ham (as found from the training set). Then, for each word provided, we multiply its probability of being spam or not, respectively. The resulting probabilities are then compared, the greater probability indicates the classification we choose.

For words that are seen in the test set and unseen in the training set, we recalculate the respective probabilities for both spam and ham of each word in the input. We do this by saying that the probability of the new word (or words) is the smoothing factor divided by the total plus the smoothing factor multiplied by the number of new words. The probabilities of the previously seen words are simply the count from before, divided once again by the total plus the smoothing factor multiplied by the number of new words.

To run the code type the command: `python q2_classifier.py -f1 spam_data/train -f2 spam_data/test -s 1` into the terminal. The command spam_data/train refers to the training data set, spam_data/test refers to the test data set, and 1 refers to the smoothing parameter.

From experimentation, we found (when the smoothing parameter is 1):

Overall Errors:
- SPAM: 554
- NOT SPAM: 0
- TOTAL WRONG: 554
- TOTAL: 1000
- TOTAL ACTUAL SPAM: 580
- TOTAL ACTUALLY NOT SPAM: 420

From this, we clearly see a significant amount of spam data points get classified as not spam, 554. This is over half of the total number of spam, and almost all of the total amount of spam values (580). However, we see that none of the not spam values are misclassified.

Altering the smoothing constant to 5, we observe:
Overall Errors:
- SPAM: 554
- NOT SPAM: 0
- TOTAL WRONG: 554
- TOTAL: 1000
- TOTAL ACTUAL SPAM: 580
- TOTAL ACTUALLY NOT SPAM: 420

Altering the smoothing constant to 20, we observe:
Overall Errors:

SPAM: 554
NOT SPAM: 0
TOTAL WRONG: 554
TOTAL: 1000
TOTAL ACTUAL SPAM: 580
TOTAL ACTUALLY NOT SPAM: 420

Altering the smoothing constant to 200, we observe:
Overall Errors:

SPAM: 557
NOT SPAM: 0
TOTAL WRONG: 557
TOTAL: 1000
TOTAL ACTUAL SPAM: 580
TOTAL ACTUALLY NOT SPAM: 420

We note that increasing the smoothing constant increases the amount of misclassified emails. However, it does so in a relatively small manner. Between 5 and 200, we only notice a difference of 3 miss classified emails.

Extra Credit:

While looking at the data provided for this assignment, I came up with some features that can hopefully add to the accuracy of this classification algorithm. The features I suggest are listed

1. Based on the number of money oriented or urgency keywords (ex. Money, price, cost, loan, gold, now, urgent, congratulations, limited, buy etc.)
2. Based on the number of misspelled words.
3. Based on the number of words with both letters and numbers.
4. Based on the number of single letters.

All of these presumably add to the classification ability.

However, I chose to implement the feature related to having both letters and numbers present. In order to do such, I counted the number of times this occurred in both spam and not spam words. The probabilities used were: pr(spam|number present), pr(not spam|number present), pr(spam|no number present), and pr(not spam|no number present). Based on whether or not I saw a number present on the test data, I multiplied the spam/not spam value (calculated as done in the before part) by their respective probabilities.

In order to run this, the command into the terminal is: `python q2_classifier.py -f1 spam_data/train -f2 spam_data/test -s 1 -f number` , where f is for feature and "number" represents the objective to look for numbers.

The result I obtained were:

Overall Errors:
- SPAM: 578
- NOT SPAM: 0
- TOTAL WRONG: 578
- TOTAL: 1000
- TOTAL ACTUAL SPAM: 580
- TOTAL ACTUALLY NOT SPAM: 420

Similarly, I did another classifier based on money oriented and urgency key words. To run this, use the command: `python q2_classifier.py -f1 spam_data/train -f2 spam_data/test -s 1 -f keywords`
The obtained results were:

Overall Errors:
- SPAM: 578
- NOT SPAM: 0
- TOTAL WRONG: 578
- TOTAL: 1000
- TOTAL ACTUAL SPAM: 580
- TOTAL ACTUALLY NOT SPAM: 420

We can see that these classifiers didn't do an extremely great job. However, it should be noted that these are two common additional classifiers used in spam detection. They will most likely provide benefits with a larger sample size.