CS534 — Implementation Assignment 2 — Due May 9th in class, 2012

General instruction.

1. You are strongly recommended to use Matlab for your implementation. However, Java, C/C++, R are also accepted.
2. You can work solo or in team of 2 people. Each team will only need to submit one copy of their solution.
3. Your source code will be submitted through the TEACH site (Due at noon on the due date).

`https://secure.engr.oregonstate.edu:8000/teach.php?type=want_auth`

Please include a readme file, which should contain the team member information.
4. You will also need to bring a hard copy of your report to the class on the due date. Similarly, clearly indicate the team members on your report.
5. Your report should be clearly written, presented in an organized manner. Figures should be correctly labeled, with axes and legends. Be careful when printing in black and white, that your figure should still be readable. Please, provide appropriate comments and notes about your work.

Naive Bayes

In this assignment you will implement the Naive Bayes classifier for document classification with both the Bernoulli model and the Multinomial model. For Bernoulli model, a document is described by a set of binary variables, and each variable corresponds to a word in the vocabulary and represents its presence/absence. The probability of observing a document $\mathbf{x}$ given its class label y is then defined as:

$$p(\mathbf{x}|y) = \prod_{i=1}^{|V|} p_{i|y}^{x_i}(1 - p_{i|y})^{(1-x_i)}$$

where $p_{i|y}$ denotes the probability that the word i will be present for a document of class y. If $x_i = 1$, the contribution of this word to the product will be $p_{i|y}$, otherwise it will be $1 - p_{i|y}$.

For the Multinomial model, a document is represented by a set of continuous variables, and each variable also corresponds to a word in the vocabulary and represents its frequency count in the document. The probability of observing a document $\mathbf{x}$ given its class label y is defined as:

$$p(\mathbf{x}|y) = \prod_{i=1}^{|V|} p_{i|y}^{x_i}$$

Here we assume that each word in the document follows a multinomial distribution of $|V|$ outcomes and $p_{i|y}$ is the probability that a randomly selected word is word i for a document of class y. Note that $\sum_{i=1}^{|V|} p_{i|y} = 1$ for $y = 0$ and $y = 1$. . x_i is the count of the number of times that word i appeared in the document.

Your implementation need to estimate $p(y)$, and $p_{i|y}$ for $i = 1, \cdots, |V|$, and $y = 1, 0$ for both models. For estimating $p_{i|y}$ you MUST use Laplace smoothing for both types of models.

Apply your Naive Bayes classifiers to the provided 20newsgroup data. By that I mean learn your models using the training data and apply the learned model to perform prediction for the test data and report the performance.

One useful thing to note is that when calculating the probability of observing a document given its class label, i.e., $p(\mathbf{x}|y)$, it can and will become overly small because it is the product of many probabilities. As a result, you will run into underflow issues. To avoid this problem, you should operate with log of the probabilities.

Data set information: The data set is the classic 20-newsgroup data set. There are six files.

- vocabulary.txt is a list of the words that may appear in documents. The line number is word's id in other files. That is, the first word ('archive') has wordId 1, the second ('name') has wordId 2, etc.

- newsgrouplabels.txt is a list of newsgroups from which a document may have come. Again, the line number corresponds to the label's id, which is used in the .label files. The first line ('alt.atheism') has id 1, etc.
- train.label contains one line for each training document specifying its label. The document's id (docId) is the line number.
- test.label specifies the labels for the testing documents.
- train.data describes the counts for each of the words used in each of the documents. It uses a sparse format that contains a collection of tuples "docId wordId count". The first number in the tuple species the document ID, and the second number is the word ID, and the final number species the number of times the word with id wordId in the training document with id docId. For example "5 3 10" species that the 3rd word in the vocabulary appeared 10 times in document 5.
- test.data is exactly the same as train.data, but contains the counts for the test documents.

  Note that you don't have to use the vocabulary file to perform your learning and testing. It is provided to help possibly interpret the models. For example, if you find that removing the first feature can help the performance, you might want to know what word that first feature actually corresponds to in the vocabulary.

Need to report:

1. Please explain how you use the log of probability to perform classification.
2. Report the overall testing accuracy for both models.
3. For each method, report the confusion matrix C, where C_{ij} species the total number of times that a class i document is classified as class j.

Bonus exploration: For bonus points, design and test a heuristic to reduce the vocabulary size and improve the classification performance. This is intended to be open-ended exploration. There will be no punishment if the exploration leads to negative impact on performance.