Casey Patton
CS 112
HW 4 Writeup

Part II: Classify Cities

a. My classifiers results were identical to those in accuracy.txt
b. Accuracy for part b: 80%
   Mean squared error: 0.137664
c. Accuracy for part c: 65%
   Mean squared error: 0.215365

Part III: Classify tweets

a) Using the Hasselhoff data set (both training and test files), my classifier had the following results in classifying German vs English:
Summary of 41 test cases, 34 correct; accuracy = 0.829268
Mean squared error: 0.124907

b) When using the training files as both the training and the tests for German vs English, i got the following results:
Summary of 255 test cases, 225 correct; accuracy = 0.882353
Mean squared error: 0.0828818

The classifier has higher accuracy when using the same exact data for both the training and the test files. This makes sense since the words we used to classify are the same words we used to test. There is likely more variation in the words used if we use different words to classify than to test, therefore we would expect lower accuracy in that case.

c) When comparing Republican vs Democrat tweets, I received the following results:
Summary of 63 test cases, 33 correct; accuracy = 0.52381
Mean squared error: 0.252509
Analysis: The classifier wasn't very accurate for this purpose, only slightly better than a coin flip.

Part IV: Extensions

I chose to do all four extensions for the 2 percent extra credit.

Extension 1

In order to improve my classifier, I added a few features. In addition to treating all single letters as features, I treated all pairs of two letters (aa, ab, ac, ….., zz) as features and all triplets of three letters as features (aaa, aab, aac, …, zzz). This added a huge amount of new features (26^3 + 26^2 more), and had mixed results.

Here is the accuracy for my new classifier:
On Part II
a) Accuracy: 91% (increase of 12%)
b) Accuracy**:** 84% (increase of 4%)
c) Accuracy: 70% (increase of 5%)

On Part III

As labeled in part III above,
a) Accuracy: 68% (decrease of 15%)
b) Accuracy: 85% (decrease of 3%)
c) Accuracy: 55% (increase of 3%)

If you average all the increases/decreases, you get a **net 1% increase in accuracy**.

Adding all these new features is a wash. While it actually did significantly better at categorizing different cities, it did significantly worse in categorizing German vs. English. One reason this might be the case is that sequences of two or three letters might not be significant in English vs. German. This data might have actually been misleading, which caused more errors in the classifier. On the other hand, this data may have been more useful in classifying different cities.

Implementation details: I only changed one file, naive_bayes_classifier (now naive_bayes_classifier_41). Only one function was altered in that file: the constructor for the class. See the comments with prefix "For Part 1" to see what was added. I also submitted feature_41.hpp, but this is identical to feature.hpp.

Extension 2

First, I modified feature.hpp (submitted as feature_42.hpp) to reflect the addition of a new class. I had to change some of the member variables to handle 3 classes rather than 2. None of the functions needed to be changed.

In naive_bayes_classifier.hpp (submitted as naive_bayes_classifer_42.hpp), I changed the class totals member variable to handle 3 classes. I had to modify some initialization functions as well as the "classify" and "GetPosterior" functions to reflect the fact that there are 3 classes. Note that my classifier for this part does NOT work with only two classes. I've targeted it to just handle three classes only.

To test my three class classifier, I did the following test cases:
1) US cities vs. Russian cities vs. Other cities
Summary of 150 test cases, 88 correct; accuracy = 0.586667

This is not nearly as accurate as the city classifier above that only had two classes, but is still far better than a guess (~33% accuracy).

2) Democrat tweets vs Republican tweets vs German language
Summary of 76 test cases, 41 correct; accuracy = 0.539474

Again, this is far better than a guess (~33%), so this classifier actually does fairly well with three classes.

3) Democrat tweets vs Republican tweets vs Russian Cities
Summary of 113 test cases, 81 correct; accuracy = 0.716814

This case actually did surprisingly well, getting over 70% of the classifications correct. This is 37% more accurate than a random guess. Probably the reason for its heightened accuracy is

that Russian cities are easily distinguishable from English language tweets, since the language is different.

Extension 3

The plot is saved as plot_43.pdf and is in the submitted package. To make this plot, I used Russia Cities and US Cities and charted the accuracy with various amount of words. I modified Russia Cities and US Cities training data to have 0 words, 25 words, 50 words, 75 words, and all 100 words, and charted the accuracy of guessing the next 50 cities at each interval. Interestingly, the accuracy when only given 10 words is not far off from the accuracy of 100 words (only a 9% difference).

Extension 4

One interesting application for a classifier is determining whether someone's writing style is more like one author or another. For example, someone could write an essay, and determine whether their writing style is more like JK Rowlings or R.L. Stine! I decided to investigate whether a classifier could be useful for this. In order to classify people's writing styles, I compared movie reviews by Roger Ebert with movie reviews by LA Times' Betsy Sharkey.

As training data, I took each person's review of "Men In Black III" (class1Training.txt and class2Training.txt). As testing data, I then took both of their reviews for "The Dictator" (class1Testing.txt and class2Testing.txt).

Results (Ebert vs Sharkey):
Summary of 19 test cases, 9 correct; accuracy = 0.473684
Mean squared error: 0.254815

Unfortunately, this wasn't very accurate. This is probably because not much about someone's writing style can be determined by the make-up of different characters they used.