COMS 4771: Machine Learning
Homework 1, due February 9.

Note: In your write-up please only provide the specific information requested. In particular, no code need be submitted.

A *Naïve Bayes Classifier* is a specific way of constructing plugin classifiers which is quite useful in practice. We assume that data are described in terms of d-dimensional feature vectors $\boldsymbol{x} = (x_1, x_2, \ldots, x_d)$. Thus, for a labeled example $(X, Y) \in \mathcal{X} \times \mathcal{Y}$ drawn from the unknown data distribution P, the random variable X can be described as a random vector $X = (X_1, X_2, \ldots, X_d)$. In the Naïve Bayes Classifier, the naïve assumption made is that the d features are *independent* of each other conditioned on the class; i.e. for any label $y \in \mathcal{Y}$, and any values of the d features $(x_1, x_2, \ldots, x_d)$, we have

$$\Pr[(X_1, X_2, \ldots, X_d) = (x_1, x_2, \ldots, x_d) \mid Y = y] \;=\; \prod_{i=1}^{d} \Pr[X_i = x_i \mid Y = y].$$

A Naïve Bayes Classifier is the plugin classifier obtained when a specific parametric model is assumed for each class conditional feature distribution and performing maximum likelihood estimation for each feature separately.

Consider a setting where we have *binary* features, i.e. all x_i are in $\{0, 1\}$. We will assume that the class conditional distribution of each x_i is Bernoulli.

Part 1 of assignment: Briefly describe the procedure and formulas for computing a Naïve Bayes Classifier for binary features with Bernoulli class conditional distributions.

Next, download the data file `usps.mat` from Canvas and load it into `MATLAB`. This will generate two matrices: `train_usps`, the training set, with 10000 labeled examples, and `test_usps`, the test set, with 1000 labeled examples. The examples represent 16×16 images of the handwritten digits from 0 to 9. Thus each example has 256 features, corresponding to the 256 pixels, and each feature is binary corresponding to whether the pixel is black or white. Each matrix has 257 columns, and each row corresponds to a labeled example. In each row, the first 256 coordinates are the binary features for the image, and the 257-th feature is the label, which is one of $\{0, 1, 2, \ldots, 9\}$. If you want to view the i-th image in the training data, use

```
imagesc(reshape(train_usps(i, 1:256), 16, 16));
```

The colors may be a little jarring; try the following to fix that:

```
colormap(1-gray);
```

Write a MATLAB function

```
function params = hw1_train_NB(train_data)
```

which computes the parameters for a Naïve Bayesian Classifier using maximum likelihood estimation with a Bernoulli model class for each feature. The output `params` can be any MATLAB data structure that works, no specific format is desired. Use this function to compute the parameters for the classifier with `train_data = train_usps`.

Write another function which computes predictions for test data given the parameters computed by the `hw1_train_NB` function, and computes the total number of errors made on the test data, with the following signature:

```
function loss = hw1_test_NB(params, test_data)
```

Part 2 of assignment: Run this function with `params` computed previously and `test_data = train_usps` to compute the training error, and with `test_data = test_usps` to compute test error, and report both.

Write another function which computes predictions for test data using a k-nearest neighbor classifier with the ℓ_2 norm using training data, and outputs the total number of errors made on the test data, as follows:

```
function loss = hw1_kNN(k, train_data, test_data)
```

In order to speed computation, you may find it useful to *vectorize* your code as much as possible: see `http://www.mathworks.com/help/matlab/matlab_prog/vectorization.html`.

Part 3 of assignment: Run `hw1_NN` with `k = 1, 3, 5`, `train_data = train_usps` and `test_data = train_data(1:1000, :)` to compute the training error on the first 1000 training examples, and report it for each value of k. Repeat the computation with `test_data = test_usps` to compute test error, and report it for each value of k.