CSE 455/555 Spring 2012 Homework 3

Jason J. Corso
TAs: Shujie Liu and Suxin Guo
Computer Science and Engineering
SUNY at Buffalo
jcorso@buffalo.edu
Date Assigned 10 April 2012
Date Due 26 April 2012
Homework must be submitted by midnight of the due-date, electronically (see below). No late work will be accepted.

Remember, you are permitted to discuss this assignment with other students in the class (and not in the class), but you must write up your own work from scratch.
*I am sure the answers to some or all of these questions can be found on the internet. Copying from **any** another source is indeed cheating. Obviously, it will undermine the primary purpose you are taking this course: to learn. This class has a zero tolerance policy toward cheaters and cheating. Don't do it.*

Problem 1: No Free Lunch Theorem (33%)

This is problem 1 from chapter 9 DHS.

One of the "conservation laws" for generalization states that the positive generalization performance of an algorithm in some learning situations must be offset by the negative performance elsewhere. Consider a very simple learning algorithm that seems to contradict this law. For each test pattern, the prediction of the *majority learning algorithm* is merely the category most prevalent in the training data.

1. Show that averaged over all two-category problems of a given number of features, the off-training set error is 0.5.
2. Repeat (1) by for the *minority learning algorithm*, which always predicts the label of the category *least* prevalent in the training data.
3. Use your answers from (1) and (2) to illustrate part 2 of the No Free Lunch Theorem.

Programming Problem: Dimension Reduction and Classification (67%)

As we saw in the case of faces, principal component analysis provides a way of creating an optimal low-dimensional representation of a dataset. Now, let's do such a PCA analysis on handwritten digits.

Refer to homework 1 for details on the MNIST digit data (for getting it, for loading it into our Python environment, and for accessing specific images). You are to use all of the 5 digits provided and observe the training and testing splits (do all component analysis and classifier training on the training set and only test on the testing set). You may use a subset of the data (say 100 samples of each digit) if memory usage becomes an issue on your terminal.

This programming assignment is different than the others: you will not be given any specific skeleton (as you have already seen them twice and you are now ready to do that on your own). Instead, you will be asked to implement a sequence of functions. Each of these should be implemented within one python file called "homework3.py" that should also have a "main" section (at the end), which will execute the entire set of functions in order listed below and generate the full output (except your written discussions, which are to be provided in a separate document).

1. Write a function to perform PCA on a group of images. This will require you to vectorize the images (i.e., do not do IMPCA). Input the number of dimensions k you want to estimate and output the set of eigenvectors and their corresponding eigenvalues (for the largest k).

2. Use the PCA function from question 1 to compute the Digit-X-Space where X is each of the six digits (separately). Plot the mean image and then the first 10 eigenvectors of each space (as images). Plot the eigenvalues (in decreasing order) as a function of dimension (for the first 100 dimensions). Describe what you find in all 12 plots.

3. Use the PCA function from question 1 to compute the Digit-Space now with all of the training data at once. Plot the mean image and then the first 20 eigenvectors (as images). Plot the eigenvalues (in decreasing order) as a function of dimension (for the first 100 dimensions). Compare and contrast what you find in this plots to the ones you created in questions 2.

4. Implement a nearest-neighbor (NN) classifier to input a training dataset, compute the PCA space (for the digit-space as a whole and not the specific digit-X-spaces), and then take a query image and assign it the class of its nearest neighbor in the PCA space.

5. Use the NN classifier to classify the testing images? Prepare a figure that shows 5 correctly classified images of each class and 5 incorrectly classified images of each class. Prepare a table giving the quantitative results over all of the testing data. Explain your findings.

6. Implement the Multiple-Class Fisher Linear Discriminant over the six digits. Train it on the training images. And use it to classify the testing images. Prepare a similar table to that in the last question. Explain and compare your findings.

7. (EXTRA 20%, NOT A REQUIRED QUESTION) Take your random forest classifier from problem one and use it to classify the six digits in the reduced PCA space. You will need to define a new feature that incorporates functions single and multiple coordinates in the PCA space. Again, prepare a testing output table and compare your findings.

8. (EXTRA 30%, NOT A REQUIRED QUESTION) Take the SVM classifier from problem two and use it to classify the six digits in the reduced PCA space. You need to change the formulation of the SVM to include a slack-term (which amounts simply to changing a subset of the constraints). Again, prepare a testing output table and compare your findings.

Submission and Grading Information

You will be required to submit the assignment in electronic form only via the CSE department `submit` script. Information will be posted on the course website when available.

The non-programming problems are graded with partial credit for accuracy and completeness.

The programming problems are also graded with partial credit. If the programs that you provide do not run to completion (this is Python and there is no compilation), the maximum amount of credit you can get is 25% of the available points, which will be awarded for correctness of the code and any discussion. Otherwise, the programming assignments are weighted equally between correctness of the code, accuracy of the output, and written discussion/solution.
