Project Part 2 [10 points] KNN Classification
(Due August 9)

In this part, you need to implement the k-Nearest Neighbor algorithm for classifying the subset of MNIST defined in Part 1 (images for digit “0” and digit “1”).

Choose the number of neighbors, k = 1, 3, 5, 7, 9, 11, 13, 15 and compute the classification accuracy for the testing set. Save your results and plot the accuracy vs. the number of neighbors.

[Note: You can also evaluate the classification accuracy for the training set, if you ignore the best matched sample. The best match for a sample in the training set will be itself, and thus the NN rule is always 100% accurate for the training samples, if we do not ignore the best matched sample.]

You will use the same features for representing the images as you did in Part 1.

Detailed Tasks (required):

1. Write code to implement the kNN algorithm (reusing the features you extracted in Part 1), and apply it on the testing set.
2. Submit a short report summarizing the results (including the accuracy as mentioned above, and the running time of your algorithm).

Detailed Tasks (optional):

1. Repeat the experiment by evaluating the training set as well, with the afore-mentioned trick of ignoring the best-matched sample.
2. Consider doing multi-class classification (e.g., considering three or four digits).

Optional tasks are to be explored on your own if you are interested and have extra time for them. No submission is required on the optional tasks. No grading will be done even if you submit any work on the optional tasks. No credit will be assigned to them even if you submit them. (So, please do not submit any work on optional tasks.)

Evaluation criteria: Working code; Correct final results.