DSC 255: Machine learning

Week 1 — Programming lab

Before attempting these problems, make sure that Python 3 and Jupyter are installed on your computer. Supporting files are in the archive `nearest-neighbor.zip`*, available from the course website.*

1. *Nearest neighbor on MNIST.* The Jupyter notebook `nn-mnist.ipynb` (in the archive file mentioned above) implements a basic 1-NN classifier for a subset of the MNIST data set. It uses a separate training and test set. Begin by going through this notebook, running each segment and taking care to understand exactly what each line is doing.

   Now do the following.

   (a) For test point 100, print its image as well as the image of its nearest neighbor in the training set. Put these images in your writeup. Is this test point classified correctly?

   (b) The *confusion matrix* for the classifier is a 10×10 matrix N_{ij} with $0 \leq i, j \leq 9$, where N_{ij} is the number of test points whose true label is i but which are classified as j. Thus, if all test points are correctly classified, the off-diagonal entries of the matrix will be zero.

   - Compute the matrix N for the 1-NN classifier and print it out.
   - Which digit is misclassified most often? Least often?

   (c) For each digit $0 \leq i \leq 9$: look at all training instances of image i, and compute their mean. This average is a 784-dimensional vector. Use the `show_digit` routine to print out these 10 average-digits.

2. *Classifying back injuries.* In this problem, you will use nearest neighbor to classify patients' back injuries based on measurements of the shape and orientation of their pelvis and spine.

   The data set contains information from 310 patients. For each patient, there are: six numeric features (the x) and a label (the y): 'NO' (normal), 'DH' (herniated disk), or 'SL' (spondilolysthesis). We will divide this data into a training set with 250 points and a separate test set of 60 points.

   - Make sure you have the data set `spine-data.txt`. You can load it into Python using the following.

     ```
     import numpy as np
     # Load data set and code labels as 0 = 'NO', 1 = 'DH', 2 = 'SL'
     labels = [b'NO', b'DH', b'SL']
     data = np.loadtxt('spine-data.txt', converters={6:  lambda s:  labels.index(s)})
     ```

     This converts the labels in the last column into 0 (for 'NO'), 1 (for 'DH'), and 2 (for 'SL').

   - Split the data into a training set, consisting of the *first* 250 points, and a test set, consisting of the remaining 60 points.

- Code up a nearest neighbor classifier based on this training set. Try both ℓ_2 and ℓ_1 distance. Recall that for $x, x' \in \mathbb{R}^d$:

$$\|x - x'\|_2 = \sqrt{\sum_{i=1}^{d} (x_i - x_i')^2}$$

$$\|x - x'\|_1 = \sum_{i=1}^{d} |x_i - x_i'|$$

Now do the following exercises, to be turned in.

(a) What error rates do you get on the test set for each of the two distance functions?

(b) For each of the two distance functions, give the *confusion matrix* of the NN classifier. This is a 3×3 table of the form:

	NO	DH	SL
NO			
DH			
SL			

The entry at row DH, column SL, for instance, contains the number of test points whose correct label was DH but which were classified as SL.