DSC 255: Machine learning

Solutions to HW 1

1. *Casting an image into vector form.* A 10×10 greyscale image has 100 coordinates with 1 pixel per coordinate. Thus the corresponding vector has dimension $d = 100$.

2. *The length of a vector.* Say $x \in \mathbb{R}^d$ where $x_i = 1$ for $i = 1, \ldots, d$. Then by our Euclidean distance formula

$$\|x\| = \sqrt{\sum_{i=1}^{d} x_i^2} = \sqrt{\sum_{i=1}^{d} 1} = \sqrt{d}$$

3. *Euclidean distance.* $\sqrt{8}$.

4. *Accuracy of a random classifier.*

   (a) If there are four labels, then no matter what the correct label is, a random classifier has exactly a 25% chance of choosing it. Therefore it has an error rate of 75%.

   (b) The best constant classifier is the one that always returns label A. It is wrong whenever the label isn't A, which occurs 50% of the time. Thus the classifier that always returns label A has error rate 50%.

5. We can work out the distances from the query to all the points.

Training point	Distance to query	label
(2,2)	$\sqrt{8.5}$	`star`
(2,4)	$\sqrt{2.5}$	`square`
(2,6)	$\sqrt{4.5}$	`star`
(4,2)	$\sqrt{6.5}$	`square`
(4,4)	$\sqrt{0.5}$	`star`
(4,6)	$\sqrt{2.5}$	`square`
(6,2)	$\sqrt{12.5}$	`square`
(6,4)	$\sqrt{6.5}$	`square`
(6,6)	$\sqrt{8.5}$	`star`

   (a) The closest point to the query is (4,4). So the point will be classified as `star`.

   (b) The 3 closest points to the query are (4,4), (2,4), and (4,6). So the point will be classified as `square`.

   (c) The 4 closest points to the query are (4,4), (2,4), (4,6), and (2,6). These are split 50/50 between `star` and `square`. The next closest point is a tie between (4,2) and (6,4). However, since both of these have the same label (`square`), the 5-NN classifier will label the query `square` no matter how it breaks ties.

6. In 4-fold cross-validation, we evenly divide our data set into 4 subsets. We hold out one subset and train on the rest. In our case, this means each time we train we will do so with 7,500 data points.

7. For 1-NN, the LOOCV procedure will misclassify the two right points. Thus the LOOCV error for 1-NN will be 50%.

   For 3-NN, the LOOCV procedure will always label the test point +. Thus the LOOCV error for 3-NN will be 25%.

8. *Nearest neighbor on MNIST.*

   (a) Training image #4711 was test point 100's nearest neighbor, and had class 4, meaning this test point was classified correctly. See Figure 1.

(a) Test Point #100 (b) Train Point #4711

Figure 1: Test Point #100 and its Nearest Neighbor

   (b) Here is some code for computing the confusion matrix:

```
confusion = np.zeros((10,10))
for i in range(0,1000):
    confusion[test_labels[i],test_predictions[i]] += 1
```

   The confusion matrix is as follows. Digit 1 was misclassified the least often and digit 9 was misclassified the most often.

$$\begin{bmatrix} 99 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 100 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 94 & 1 & 0 & 0 & 0 & 3 & 1 & 0 \\ 0 & 0 & 2 & 91 & 2 & 4 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 97 & 0 & 0 & 0 & 0 & 3 \\ 1 & 0 & 0 & 0 & 0 & 98 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 99 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 & 1 & 0 & 0 & 94 & 0 & 1 \\ 2 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 92 & 1 \\ 1 & 1 & 1 & 1 & 2 & 1 & 0 & 3 & 0 & 90 \end{bmatrix}$$

   (c) Here is code for computing the average of each digit and displaying it.

```
for digit in range(0,10):
    I = (train_labels==digit)  # Find indices of training points with label 'digit'
    avg = np.mean(train_data[I,:],axis=0) # Compute the mean of these points
    show_digit(avg) # Display the mean
```

The following figures contain the mean image for every digit.