

# Introduction to Machine Learning, Fall 2014 - Exercise session III

Rodion “rodde” Efremov, student ID 013593012

November 12, 2015

## **Problem 1 (3 points)**

## **Problem 2 (3 points)**

## **Problem 3 (3 points)**

## **Problem 4 (9 points)**

**(a)**

Download the MNIST data from the [course web page](#). In addition to the actual data, the package contains some functions for easily loading the data into Python/Matlab/Octave/R and for displaying digits. See the README files for details. Load the first  $N = 5,000$  images using the provided function.

## **Solution to (a)**

Check!

**(b)**

Use the provided functions to plot a random sample of 100 handwritten digits, and show the associated labels. Verify that the labels match the digit images. (This is a sanity check that you have the data [is] in the right format.)

**Solution to (b)**

|   |   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 2 | 3 | 9 | 2 | 0 | 6 | 6 | 8 | 6 |
| 9 | 4 | 7 | 5 | 4 | 0 | 4 | 6 | 3 | 8 |
| 7 | 2 | 4 | 5 | 8 | 1 | 1 | 8 | 7 | 6 |
| 9 | 5 | 4 | 6 | 3 | 1 | 3 | 4 | 9 | 3 |
| 0 | 5 | 4 | 2 | 2 | 7 | 8 | 7 | 5 | 4 |
| 5 | 6 | 3 | 7 | 2 | 8 | 9 | 0 | 5 | 9 |
| 0 | 9 | 8 | 1 | 6 | 9 | 3 | 6 | 5 | 8 |
| 9 | 9 | 6 | 8 | 4 | 7 | 2 | 4 | 4 | 4 |
| 7 | 8 | 0 | 0 | 8 | 7 | 8 | 2 | 0 | 8 |
| 6 | 3 | 0 | 1 | 8 | 7 | 3 | 8 | 2 | 1 |

The labels are:

```

0 2 3 9 2 0 6 6 8 6
9 4 7 5 4 0 4 6 3 8
7 2 4 5 8 1 1 8 7 6
9 5 4 6 3 1 3 4 9 3
0 5 4 2 2 7 8 7 8 4
5 6 3 7 2 8 9 0 5 9
0 9 8 1 6 9 3 6 5 8
9 9 6 8 4 7 2 4 4 4
7 8 0 0 8 7 8 2 0 8
6 3 0 1 8 7 3 8 2 1

```

**(c)**

Divide the data into two parts: A ‘training set’ consisting of the first 2,500 images (and associated labels), and a ‘test set’ containing the remaining 2,500 images (and their associated labels).

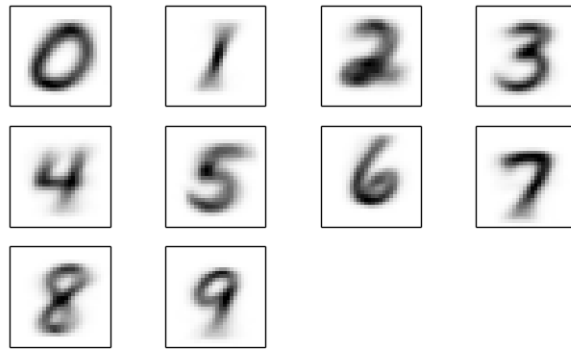
**Solution to (c)**

Check!

**(d)**

For each of the ten classes (digits 0-9), compute a class *prototype* given by the mean of all the images in the training set that belong to this class. That is, select from the training set all images of class '0' and compute the mean image of these; this should look sort of like a zero. Do this for all ten classes, and plot the resulting images. Do they look like what you would expect?

### Solution to (d)



Obviously, the prototypes look good.

### (e)

For each of the images in the test set, compute the Euclidean distance of the image to all 10 prototypes, and classify the test image into the class for which the distance to the prototype is the smallest. So, if a test image is closer to the prototype for '3' than it is to the prototypes for any of the other digits, predict its class to be '3'. Compute and display the resulting *confusion matrix*.

### Solution to (e)

The confusion matrix for the prototype-based classifier is

|     |     |     |     |     |    |   |   |    |    |
|-----|-----|-----|-----|-----|----|---|---|----|----|
| 228 | 0   | 1   | 2   | 0   | 4  | 6 | 2 | 2  | 0  |
| 0   | 281 | 1   | 0   | 0   | 3  | 0 | 0 | 1  | 0  |
| 2   | 19  | 192 | 5   | 5   | 2  | 2 | 2 | 11 | 1  |
| 1   | 9   | 4   | 193 | 2   | 20 | 0 | 5 | 11 | 8  |
| 1   | 4   | 1   | 0   | 202 | 0  | 5 | 2 | 0  | 40 |

|   |    |    |    |    |     |     |     |     |     |
|---|----|----|----|----|-----|-----|-----|-----|-----|
| 9 | 16 | 1  | 14 | 11 | 136 | 2   | 6   | 1   | 11  |
| 5 | 12 | 15 | 0  | 9  | 8   | 198 | 0   | 0   | 0   |
| 1 | 14 | 1  | 0  | 15 | 0   | 0   | 232 | 1   | 11  |
| 3 | 13 | 9  | 25 | 4  | 19  | 1   | 1   | 151 | 14  |
| 3 | 4  | 6  | 2  | 38 | 2   | 1   | 14  | 2   | 179 |

(f)

Classify each of the test images with a nearest neighbor classifier: For each of the test images, compute its Euclidean distance to all (2,500) of the training images, and let the predicted class be the class of the closest training image. Compute and display the resulting confusion matrix.

### Solution to (f)

The confusion matrix of the kNN-classifier is

|     |     |     |     |     |     |     |     |     |     |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 241 | 0   | 0   | 2   | 0   | 1   | 1   | 0   | 0   | 0   |
| 0   | 283 | 0   | 0   | 1   | 1   | 0   | 1   | 0   | 0   |
| 2   | 6   | 216 | 5   | 0   | 1   | 1   | 5   | 5   | 0   |
| 0   | 0   | 3   | 221 | 2   | 11  | 2   | 4   | 5   | 5   |
| 0   | 3   | 0   | 0   | 231 | 0   | 0   | 2   | 0   | 19  |
| 5   | 3   | 0   | 6   | 0   | 183 | 5   | 1   | 1   | 3   |
| 3   | 1   | 0   | 0   | 1   | 1   | 240 | 0   | 0   | 1   |
| 0   | 3   | 2   | 0   | 6   | 0   | 0   | 257 | 0   | 7   |
| 4   | 5   | 9   | 10  | 1   | 8   | 0   | 2   | 196 | 5   |
| 2   | 2   | 0   | 2   | 12  | 1   | 2   | 11  | 0   | 219 |

(g)

Compute and compare the error rates of both classifiers (the prototype-based classifier and the nearest neighbor classifier). Which is working better? Based on the confusion matrix, which digits are confused with each other? Why do you think this is?

### Solution to (g)

The error rate of the prototype-based classifier is 0.2032 and the error rate of kNN is only 0.852. Obviously, kNN surpasses prototype-based classifier. It seems that 4 and 9, 3 and 8 are problematic. Obviously, the aforementioned digits in each pair share a lot of common geometry.