DSE 220: Machine learning

Worksheet 1 — Nearest neighbor

Nearest neighbor classification

1. *Casting an image into vector form.* A 10×10 greyscale image is mapped to a d-dimensional vector, with one pixel per coordinate. What is d?

2. *The length of a vector.* The Euclidean (or L_2) length of a vector $x \in \mathbb{R}^d$ is

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

   where x_i is the ith coordinate of x. This is the same as the Euclidean distance between x and the origin. What is the length of the vector which has a 1 in every coordinate? Your answer may be a function of d.

3. *Euclidean distance.* What is the Euclidean distance between the following two points in $\mathbb{R}^3$?

$$(1, 2, 3), \quad (3, 2, 1)$$

4. *Euclidean distance.* What is the Euclidean distance between the following two points $x, x' \in \mathbb{R}^d$?

   - x has all coordinates equal to 1.
   - x' has all coordinates equal to -1.

5. *Accuracy of a random classifier.* A particular data set has 4 possible labels, with the following frequencies:

Label	Frequency
A	50%
B	20%
C	20%
D	10%

   (a) What is the error rate of a classifier that picks a label (A,B,C,D) at random, each with probability 1/4?

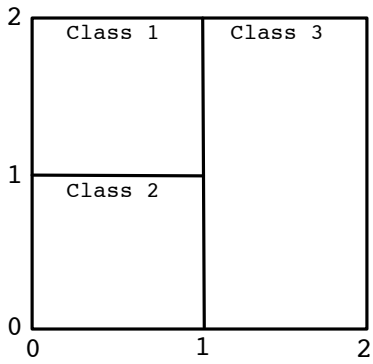
   (b) One very simple type of classifier just returns the same label, always.
   - What label should it return?
   - What will its error rate be?

6. *Decision boundary of the nearest neighbor classifier.* In this problem,

   - The data space is $\mathcal{X} = [0, 2]^2$: each point has two coordinates, and they lie between 0 and 2.

- The labels are $\mathcal{Y} = \{1, 2, 3\}$.

The correct labels in different parts of $\mathcal{X}$ are as shown below.



(a) What is the label of point $(0.5, 0.5)$?

Now suppose you have a training set consisting of just two points, located at

$$(0.5, 0.5), \ (0.5, 1.5).$$

(b) What label will the nearest neighbor classifier assign to point $(1.5, 0.5)$?

(c) What label will the nearest neighbor classifier assign to point $(2, 2)$?

(d) Which label will this classifier never predict?

(e) Now suppose that when the classifier is used, the test points are uniformly distributed over the square $\mathcal{X}$. What is the error rate of the 1-NN classifier?

7. *Programming exercise.* To begin with:

- Install Python 3 and Jupyter on your computer.
- Obtain `nn1-notebook.zip` from the course website and uncompress it.
- The Jupyter notebook `nn-mnist.ipynb` implements a basic 1-NN classifier for a subset of the MNIST data set. It uses a separate training and test set.
- Go 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?

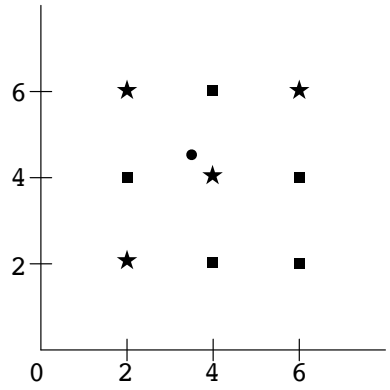
(b) The *confusion matrix* for the classifier is a 10×10 matrix N_{ij} with $0 \leq i, j \leq 9$, where N_{ij} is the number of test points whose true label is i but which are classified as j. Thus, if all test points are correctly classified, the off-diagonal entries of the matrix will be zero.

- 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.

k-nearest neighbor

8. In the picture below, there are nine training points, each with label either `square` or `star`. These will be used to guess the label of a query point at $(3.5, 4.5)$, indicated by a circle.



   Suppose Euclidean distance is used.

   (a) How will the point be classified by 1-NN? The options are `square`, `star`, or `ambiguous`.

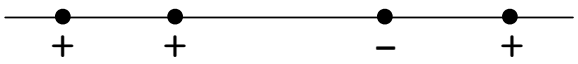
   (b) By 3-NN?

   (c) By 5-NN?

9. We decide to use 4-fold cross-validation to figure out the right value of k to choose when running k-nearest neighbor on a data set of size 10,000. When checking a particular value of k, we look at four different training sets. What is the size of each of these training sets?

10. An extremal type of cross-validation is *n-fold cross-validation* on a training set of size n. If we want to estimate the error of k-NN, this amounts to classifying each training point by running k-NN on the remaining $n - 1$ points, and then looking at the fraction of mistakes made. It is commonly called *leave-one-out cross-validation* (LOOCV).

    Consider the following simple data set of just four points:



    What is the LOOCV error for 1-NN? For 3-NN?

11. *Programming exercise.* 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.

    - Download 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 = ['NO', 'DH', '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.

Open-ended projects

12. *Prototype selection.* One way to speed up nearest neighbor classification is to replace the training set by a carefully chosen subset of "prototypes".

Think of a good strategy for choosing prototypes from the training set, bearing in mind that the ultimate goal is good classification performance. Assume that 1-NN will be used.

Then implement your algorithm, and test it on the MNIST dataset, available at:

`http://yann.lecun.com/exdb/mnist/index.html`

Submit a report containing the following elements.

- *A short, high-level description of your idea for prototype selection.*
  A few sentences should suffice. These should be crystal clear: they should communicate the key idea to the reader.

- *Concise and unambiguous pseudocode.*

  (Please do not submit any actual code.) Once again, clarity and conciseness are of the essence. Your scheme should take as input a labeled training set as well as a number M, and should return a set of M prototypes.

- *Experimental results.*

  A (clearly labeled) table or graph of results showing classification performance on MNIST for a few values of M, including $M = 100, 500, 1000, 5000, 10000$. In each case, you should compare the performance to that of uniform-random selection (that is, picking M of the training points at random). For any strategy with randomness, you should do several experiments and give error bars – give all relevant details, including the formulas you used for computing confidence intervals.

  The pseudocode and experimental details must contain all information needed to reproduce the results.

- *Critical evaluation.*

  Is your method a clear improvement over random selection? Is there further scope for improvement? What would you like to try next?