Homework 2

36-350: Data Mining

Due at the start of class, Friday, 11 September 2009

NOTICE: Except as otherwise specified, you can use any code you wrote for previous assignments, and any code I provide in the lectures, the homework assignments, or the solutions to previous assignments.

1. In a typical data mining application, a news agency wants to search through video archives and detect all frames depicting an event, e.g. fireworks at night, given some examples. You can regard a video as a sequence of images. How could this be implemented using similarity search? (Don't give code, just a brief description of the method.)

2. Recall that in **nearest-neighbor** classification, we guess that a new vector belongs to the same class as the closest perviously-seen vector whose class is known.

   (a) Write an R function to do nearest-neighbor classification. Your function should take the following inputs:

   - A p-dimensional vector to be classified
   - A data frame of labeled example vectors, in which the first column gives the class labels, and the other p columns give the components of the vectors.

   Your classifier should return the label of the example vector closest (in Euclidean distance) to the new vector. *Hint:* consider the `nearest.points` function in `01.R`.

   (b) Create a data frame for the following set of examples.

label				
red	3	10	2	11
blue	17	−17	9	−1
red	−4	9	−2	−1
blue	4	0	2	−5
blue	8	−1	6	−12
red	19	3	23	14

   Check that each of these example vectors is its own nearest neighbor, and that in that case your classifier returns the correct class label.

(c) Write a function that will take a data-frame of labeled example vectors, and evaluate the accuracy of nearest-neighbor classification by holding each of them out. This is called **leave-one-out cross-validation**.

- The only input should be the data-frame of labeled examples.
- For each row in the data frame, call your nearest neighbor classifier with that row's vector as the first input, and the *other* rows of the data frame as the labeled examples.
- Return the fraction of times the nearest-neighbor rule guesses the correct class label.

Check this by applying your function to the data frame from the previous part. It should say that the method is accurate 5/6 of the time.

(d) A simple mistake in the previous part would lead you to conclude that nearest neighbor classification is *always* 100% accurate on *any* data set. Describe the mistake.

(e) Use leave-one-out cross-validation to find the accuracy of the nearest-neighbor rule on the art/music news stories from the last assignment. (You should include both IDF weighting and Euclidean length normalization.)

3. In **prototype** classification, we represent each class by the average of the vectors belonging to that class, and assign new vectors to the class whose prototype is closest.

(a) Write a function which takes a matrix where each row is a vector, and finds the mean vector. (Remember that you find the mean vector by taking the means of all the components.) The input of the function should be an $n \times p$ matrix of numbers, and the output a p-dimensional vector of numbers. Check that your function works on example matrices with one, two and three rows.

(b) Write a function which finds the mean vectors for all the classes in a data frame of labeled examples. The first column of the input should be the class label, and the remaining columns should be the vector components. The output should be another data frame with the same format, but should have only one row per class. Check your code by calculating the means for the data frame in the previous problem; the answer should be

label				
red	6	$7\frac{1}{3}$	$7\frac{2}{3}$	8
blue	$9\frac{2}{3}$	-6	$5\frac{2}{3}$	-6

(c) Write a function to do prototype classification. It should have the same inputs and outputs as your function for nearest-neighbor classification. Check that if you ask your function to classify one of your

prototype vectors, it is assigned to the correct class. *Hint:* One way to write this function actually calls your nearest-neighbor function.

(d) Write a function to do leave-one-out cross-validation using the prototype method. It should have the same inputs and outputs as the function you wrote to do this for nearest-neighbor classification. How accurate is it on the six-vector example?

(e) How accurate is the prototype method on the *Times* stories?

4. (Extra Credit) The functions you wrote to do cross-validation for the nearest neighbor method and for the prototype method are likely very similar to each other. Write a *single* function which takes two arguments, a data-frame of labeled examples and an *arbitrary* classifier function, and does leave-one-out cross-validation. (You can assume that the classifier function takes the same inputs and returns the same outputs as the ones you've written, but you cannot assume it has to be one of them.) — If you do this, you do not *have* to write separate functions to handle the individual classifiers, but you may want to anyway to warm yourself up.

Note: there are a number of functions already in various R packages (e.g., `class` and `knnflex`) to do nearest-neighbor classification. Writing a function which just calls one is not acceptable[1]; however, you can use them to check whether your code is working properly.

[1] Yes, someone tried that before.