Part B - Intro to learning:

Part A - dry part:

1. Solution:

We want to classify :

* When using the Euclidean function:
* When using the Euclidean function:

1. The value k = 1 minimizes the training error for the given training set. Since each sample in the training set is a neighbor to itself then using k = 1 for classifying a sample from the training set would result to a true positive each time.

Which means the training error resulting from choosing k = 1 is 0.

1. k equals to the size of the training set gives us a majority classifier since then the k nearest neighbors are all the training set and the sample would get classified according to the most common classification among the k neighbors (all the training set) which means it would get classified according to the majority.

In the even set,

1. Using too low values of k would result to difficulty coping with noise (overfitting).

Using too high values of k would result in considering neighbors who aren’t really close nor relevant (underfitting).

1. Correct. Prove:

Let’s take

We want to classify :

So the original version of the algorithm would classify (0,0) as - since (3,3) is the nearest neighbor.

In the new variation:

* If , then the classification will be positive vacuously.
* If , then the classification will be positive due to a tie between (3,3) and (3,4).
* if , then the classification will be positive since they are 2 positive and one negative.

Thus, in all cases (for all r) the new variation will classify (0,0) as + as opposed to the classification of the original version.

1. Incorrect. Prove:

Given d, r, training set and a test example.

After calculating the distance of all the samples in the training set from the test example, we choose k equal to the number of samples in the training set which are within a radius r from the test sample (k = 0 if there are none).

We will get the same samples picked in the new variation since they are the closest.

Thus, since we are using the same sample, we’ll get the same classification.

Splitting the fun:

Solution: The claim is incorrect.

Let T be a decision tree of depth d and T’ be the tree resulting by pruning the last level of T (of depth d-1). The nodes in T checks if the value of some feature is positive or not.

Given test example where .

We’ll classify x using T’ and the normal decision rule learned in class.

Doing so, we’ll reach the most right ( always picking right since ).

Now we classify x using T and the epsilon-decision rule, we require that

We can see that up to depth d-1, we’ll always go right like in the normal decision rule. In depth d, to ensure that we always decide the same like the normal decision rule we define

and that would result in getting the same decision for both.

However, looking at the test example where

.

We decide using the epsilon-decision rule, we would not be guaranteed to get the same result as in the normal decision rule on the prune tree.

Thus, it’s impossible to find epsilon which would classify all testing examples the same as the normal decision rule.