# Homework1

**Problem 1:**

1. The algorithm used for prototype selection is inspired by the CNN (condensed nearest neighbor) algorithm.

**CNN**: Let TR be the training set and PS be the prototype set. We initialize the prototype set to be an empty set. Each instance *(i)* in TR is classified using only the instances in PS. If the instance *(i)* is misclassified, it is added to PS. This continues till there are no more elements to be classified in TR.

1. Pseudocode:

// Classification using CNN (Condensed nearest neighbor)

// Method: get\_cnn\_prototype\_set

// Input: training\_set(images, labels), M (size of the prototype\_set)

// Output: p\_set(images, labels) of size M

Initialize empty prototype set p\_set(images, labels) and r\_set(images, labels)

// r\_set contains samples which are not selected in the p\_set

Randomly shuffle the training\_set {(images, labels)}

while(training\_set) && sizeof(p\_set) < M:

sample\_set(image, label) = remove a sample from the training\_set

// calculate\_nearest\_neighbor will return the label of the nearest neighbor in the

training set

if( sample\_set[label] != **calculate\_nearest\_neighbor**(sample\_set[image], proto\_training\_set)):

Add sample\_set to p\_set

Else:

Add sample\_set to r\_set

if(sizeof(p\_set) < M)

Randomly select max((M - sizeof(p\_set)), sizeof(r\_set)) samples from r\_set and add them to the p\_set

return p\_set

// Note:

// the images are classified using 1NN

// Distance measure used = Eucledian distance = ||x-y||2

1. Comparing error rates for classification with random prototype vs. CNN prototype of size M

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Error Rates **🡪** | Random Prototype Selection | | | CNN Prototype Selection | | |
|  | Run:1 | Run:2 | Run:3 | Run:1 | Run:2 | Run:3 |
| M = 1000 | 0.114 | 0.1112 | 0.1151 | 0.1115 | 0.1146 | 0.1086 |
| M = 5000 | 0.0659 | 0.0664 | 0.0635 | 0.0615 | 0.0639 | 0.0626 |
| M = 10000 | 0.0509 | 0.0522 | 0.0528 | 0.0507 | 0.0513 | 0.0517 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| M | Random Prototype Selection | | CNN Prototype Selection | | Average Improvement |
| Mean | Variance | Mean | Variance |
| 1000 | 0.1134 | 2.67 \* 10-6 | 0.1116 | 6 \* 10-6 | 1.6% |
| 5000 | 0.0653 | 1.6 \* 10-6 | 0.0627 | 9.9 \* 10-7 | 3.98% |
| 10000 | 0.0520 | 6.3 \* 10-7 | 0.0512 | 1.7 \* 10-7 | 1.54% |

**Problem 2:**

1. The Bayes-optimal classifier:

Optimal risk:

// solving the integrals

1. From the training set the decision boundary is as follows:

True error rate of the classifier:

// Using marginalization and conditional independance

*h(x) = 0* only when *x < -0.6 and x > 0.5*

*h(x) = 1* only when

At any given x,

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| x < 0.5 | 0.8 | 0.1 \* 0.2 = 0.02 | 0 |
| -0.5<= x <= 0.5 | 0.2 | 0.1\*0.8 = 0.08 | 0 |
| x>0.5 | 0.6 | 0.1\*0.4 = 0.04 | 0 |

The decision boundary would be when



From part c,

At any given x,

**Problem 3:**

1. **L1 distance :**

Hence,

Triangle Inequality:

// Taking summation over all i

**Therefore, it is a metric**

1. **d1 +d2 where d1 d2 are both metrics**

Proof:

2.

From 1,

Which implies

3.

4.

Triangle inequality:

Adding up,

**Therefore, it is a metric**

1. **d(x, y) = # of positions on which x and y differ**

We can turn x to y by changing atmost d(x, y) characters and turn y to z by changing atmost d(y, z) characters. So to turn x into z will change no more than d(x, y) + d(y, z) characters. Hence,

**Therefore, it is a metric**

Proof by contradiction:

According to the triangle rule of inequality:

Let

Substituting values in the triangle inequality,

**Therefore, it is not a metric**

Proof by contradiction:

Given: *X* = {p ∈ Rm: pi ≥ 0, ∑i pi = 1}, Let m = 2 and p1 = 0.5 and p2 = 0.5

Y = {q ∈ Rm: qi ≥ 0, ∑i qi = 1}, Let m = 2 and q1 = 0.2 and q2 = 0.8

**Therefore, it is not a metric**