

# Homework #3

Claire Jung

May 1, 2023

In collaboration with Natasha Antipova.

## Problem 1

For linearly separable data, a decision tree can classify the data. There would be an upper bound on the branches, of  $N-1$ , as well as an upper bound on the depth of  $N-1$ .

This is because in worst case scenario, say the decision boundary is a diagonal line  $x = y$ , it would take  $N-1$  branches to separate the two classes.

## Problem 2

For non linearly separable data, a decision tree could still classify the points. There would be an upper bound on the branches, of  $N-1$ , as well as an upper bound on the depth of  $N-1$ .

Imagine positive and negative points alternating along an axis in the data, either horizontal or vertical. It would take  $N-1$  branches with a depth of  $N-1$  to classify this data.

## Problem 3

We know the weights are normalized,  $\sum_{i=1}^N W_i^T = 1$ , for any time  $T$ . Considering time  $T + 1$ ,

$$\sum_i^N W_i^{T+1} = \sum_i^N W_i^T \exp(-\alpha_{T+1} y_i h_{T+1}(x_i)) = 1$$

Split the sum for right and wrong classifications,

$$\sum_i^N \exp(-\alpha_{T+1}) W_i^T + \sum_i^N \exp(\alpha_{T+1}) W_i^T = 1$$

As the weights are normalized, and  $\epsilon_{T+1} = \sum_i^N W_i^T$  for the wrong answers.

$$\exp(-\alpha_{T+1})(1 - \epsilon_{T+1}) + \exp(\alpha_{T+1})(\epsilon_{T+1}) = 1$$

Plugging in the expression for  $\alpha$ , where  $\alpha_t = \frac{1}{2} \ln(\frac{1-\epsilon_t}{\epsilon_t})$

$$\sqrt{\frac{\epsilon_{T+1}}{1 - \epsilon_{T+1}}}(1 - \epsilon_{T+1}) + \sqrt{\frac{1 - \epsilon_{T+1}}{\epsilon_{T+1}}}\epsilon_{T+1} = 1$$

$$2\sqrt{\epsilon_{T+1} - \epsilon_{T+1}^2} = 1 \rightarrow \epsilon_{T+1} = \frac{1}{2}$$

You could not select the same classifier again, as you would simply have a zero vote and your weights would be unchanged.

$$\alpha_t = \frac{1}{2} \ln\left(\frac{1 - 1/2}{1/2}\right) = 0$$

## Problem 4

Starting off with the loss we derived in the previous problem,

$$L = \exp(-\alpha_T)(1 - \epsilon_T) + \exp(\alpha_T)(\epsilon_T)$$

$$\frac{\partial L}{\partial \alpha} = -\alpha \exp(-\alpha)(1 - \epsilon) + \alpha \exp(\alpha)\epsilon = 0$$

$$0 = \exp(-\alpha)\epsilon - \exp(-\alpha) + \exp(\alpha)\epsilon$$

$$\exp(\alpha)\epsilon = \exp(-\alpha)(1 - \epsilon)$$

$$\exp(2\alpha) = \frac{1 - \epsilon}{\epsilon}$$

$$2\alpha = \ln\left(\frac{1 - \epsilon}{\epsilon}\right)$$

$$\alpha = \frac{1}{2} \ln\left(\frac{1 - \epsilon}{\epsilon}\right)$$

## Problem 5

We can write our optimization as,

$$\operatorname{argmin} \left[ \frac{1}{2} \|w\|^2 + C \sum_i^N \xi_i \right] = J$$

where  $y^{(i)}(w^T \phi(x)^{(i)} + w_0) \geq 1 - \xi_i, i = 1, \dots, N$  and  $\xi_i \geq 0$ . Using Lagrange multipliers and the KKT theorem, this can be rewritten as

$$\operatorname{argmax}_{\alpha} \operatorname{argmin}_w \left[ \frac{1}{2} \|w(\alpha)\|^2 + \sum_i^N \alpha_i [1 - y_i(w_0(\alpha) + w(\alpha) \cdot x_i)] \right]$$

where now  $0 \leq \alpha_i \leq C$

Computing derivatives to get rid of the  $w$  terms,

$$\frac{\partial J}{\partial w} = 0 \rightarrow w = \sum_i^N \alpha_i y_i x_i$$

$$\frac{\partial J}{\partial w_0} = 0 \rightarrow 0 = \sum_i^N \alpha_i y_i$$

We now have an additional equality condition.

Plugging these all in, we have a single optimization problem over alpha

$$\operatorname{argmax}_{\alpha} \left[ -\frac{1}{2} \sum_i^N \sum_j^N \alpha_i \alpha_j y_i y_j x_i \cdot x_j + \sum_i^N \alpha_i \right] \rightarrow \operatorname{argmin}_{\alpha} \left[ \frac{1}{2} \sum_i^N \sum_j^N \alpha_i \alpha_j y_i y_j x_i \cdot x_j - \sum_i^N \alpha_i \right]$$

This can now be used in a quadratic program to solve for the  $\alpha$  terms, using  $K(\cdot, \cdot)$  as our dot product kernel.

$$H = |y\rangle\langle y| K(x, x)$$

$$f = [-1, -1, \dots, -1], \text{ vector of -1s}$$

$$B = y$$

$$b = 0$$

$$A = [[-I], [I]], \text{ where } I \text{ is an } N \times N \text{ identity matrix}$$

$$a = [[0], [C]] \text{ where } 0 \text{ is a } N \text{ column vector of 0s and } C \text{ is an } N \text{ column vector of Cs}$$