COMP9417 - Machine Learning
Tutorial: Ensemble Learning

Question 1 (Exponential Loss AdaBoost Derivation)

The goal of this question will be to understand one approach to deriving the AdaBoost algorithm. We have access to a data set:

$$\{(x_i, y_i) : i = 1, \ldots, n,\ x_i \in \mathcal{X},\ y_i \in \{-1, 1\}\},$$

the notation here implies that the features x_i take values in some set $\mathcal{X}$, for example $\mathcal{X} = \mathbb{R}^d$, and the labels y_i are either $+1$ or -1. Consider the class of weak learners:

$$\mathcal{H} = \{h : \mathcal{X} \to \{-1, 1\}\}.$$

The notation here means that each hypothesis h in $\mathcal{H}$ is a function taking values in $\mathcal{X}$ and returning values in $\{-1, 1\}$. Next, define the following exponential loss:

$$L(h, y) := L(h) = \sum_{i=1}^{n} \ell(h(x_i), y_i), \qquad \ell(h(x_i), y_i) = e^{-y_i h(x_i)}.$$

(a) Describe the behaviour of the loss function ℓ. What happens when our hypothesis returns the correct label ($h(x_i) = y_i$)? What about when an incorrect label is returned?

(b) Recall that the AdaBoost algorithm seeks to recursively construct an ensemble classifier which at the $(m-1)^{\text{st}}$ step has the form:

$$\mathsf{sign}(C_{m-1}(x)) = \begin{cases} +1 & \text{if } C_{m-1}(x) \geq 0 \\ -1 & \text{if } C_{m-1}(x) < 0, \end{cases}$$

where

$$C_{m-1}(x) = \alpha_1 h_1(x) + \alpha_2 h_2(x) + \cdots + \alpha_{m-1} h_{m-1}(x),$$

and the weights $\alpha_1, \ldots, \alpha_{m-1}$ are all positive. Assume that you have access to C_{m-1}, so that you know $\alpha_1, \ldots, \alpha_{m-1}$ and $h_1, \ldots, h_{m-1}$. The two challenges at this stage are then:

- How do we pick a new hypothesis h at step m?
- How do we pick the corresponding weight α_m?

Show that

$$L(C_m) = \sum_{i=1}^{n} w_i^{(m)} e^{-\alpha_m y_i h_m(x_i)}$$

where

$$w_i^{(m)} := e^{-y_i C_{m-1}(x_i)}.$$

Explain why these weights $w_i^{(m)}$ for $i = 1, \ldots, n$ are larger for data points on which C_{m-1} made a mistake, and smaller for data points that were correctly classified by C_{m-1}.

(c) Show further that

$$L(C_m) = e^{-\alpha_m} \sum_{i:h_m(x_i)=y_i} w_i^{(m)} + e^{\alpha_m} \sum_{i:h_m(x_i)\neq y_i} w_i^{(m)}.$$

Note that the first sum is only summing over indices where the base learner h_m does not make a mistake, and the second sum is summing over indices on which a mistake is made. We can interpret the first sum as the component of the error due to making correct classification, and the second sum as the component due to misclassification.

(d) Since we want to find α_m, h_m, it makes sense to minimize this loss with respect to these parameters, that is, we want to find the base learner and weight that solve

$$\min_{\alpha_m>0, h_m\in\mathcal{H}} L(C_m).$$

Note that we only want to allow positive weights, and only hypotheses that belong to our set of weak learners. Argue heuristically that in order to minimize the loss, we should always pick h_m to minimize the (weighted) classification error, i.e. the best possible $h_m \in \mathcal{H}$ satisfies

$$h_m^* = \arg\min_{h_m\in\mathcal{H}} \sum_{i:h_m(x_i)\neq y_i} w_i^{(m)}.$$

Explain why this choice of base learner can be interpreted as: trying to learn the mistakes of the previous learners.

(e) Suppose that we set $h_m = h_m^*$. What remains is to choose α_m. Solve the problem

$$\arg\min_{\alpha_m>0} L(C_m^*, y) = \arg\min_{\alpha_m>0} L(C_{m-1} + \alpha_m h_m^*, y)$$

by differentiating with respect to α_m and setting equal to zero. Show that the solution is

$$\alpha_m^* = \frac{1}{2}\ln\left(\frac{\sum_{i:h_m^*(x_i)=y_i} w_i^{(m)}}{\sum_{i:h_m^*(x_i)\neq y_i} w_i^{(m)}}\right).$$

(f) Define the weighted classification error proportion:

$$\epsilon_m := \frac{\sum_{i:h_m(x_i)\neq y_i} w_i^{(m)}}{\sum_{i=1}^n w_i^{(m)}}.$$

Show that

$$\alpha_m^* = \frac{1}{2}\ln\left(\frac{1-\epsilon_m}{\epsilon_m}\right).$$

Explain why α_m^* is always positive, and further clarify the role played by α_m^*

(g) Finally, we need to formalize the change in weights at each iteration. Show that the weight updates can be written in the following recursive form:

$$\begin{cases} w_i^{(m)} &= w_i^{(m-1)} \sqrt{\frac{\epsilon_{m-1}}{1-\epsilon_{m-1}}} & \text{if} \quad h_{m-1}(x_i) = y_i, \\ w_i^{(m)} &= w_i^{(m-1)} \sqrt{\frac{1-\epsilon_{m-1}}{\epsilon_{m-1}}} & \text{if} \quad h_{m-1}(x_i) \neq y_i, \end{cases} \qquad i = 1, \ldots, n,$$

and provide an interpretation of these weight updates.

(h) Often, it is desirable to have the weights at each iteration sum to 1. This allows us to more easily interpret the weights of each point. We can easily achieve this by adding a normalization stem at each iteration in which we divide all weights by the normalization constant:

$$Z_m = \sum_{i=1}^{n} w_i^{(m)}.$$

Assuming that the weights from the previous iteration $(m-1)$ are already normalized, show that

$$Z_m = 2\sqrt{\epsilon_{m-1}(1-\epsilon_{m-1})}.$$

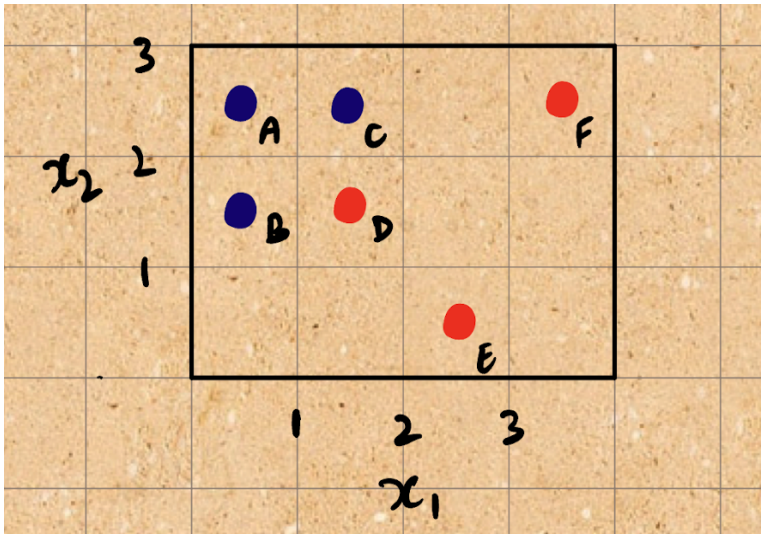
(i) The normalization step requires us to use the normalized weights:

$$\frac{w_i^m}{Z_m}.$$

What are the explicit weight updates (use part (g)) in the normalized case?

Question 2. AdaBoost Toy Example

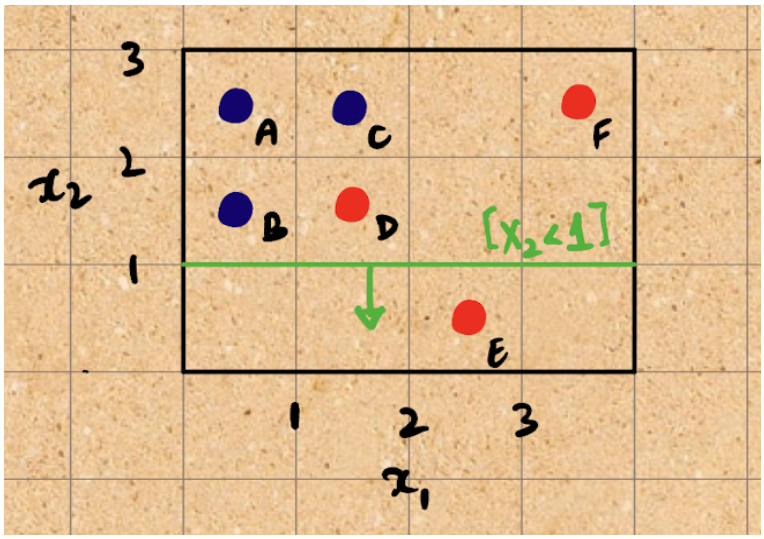
We will now implement AdaBoost by hand on a toy example. Consider the following two-dimensional points labelled A, B, C, D, E, F. The blue points have true label -1 and the red points have true label $+1$. The size of each of the points is equal since we are starting out with uniform weights, $w_i^{(1)} = \frac{1}{6}$ for $i = 1, \ldots, 6$.



We consider the following class of simple hypotheses:

$$\mathcal{H} = \{[x_1 > 1], [x_1 > 2], [x_1 > 3], [x_2 < 1], [x_2 < 2]\}.$$

In the following plot, we see an example of what the hypothesis $[x_2 < 1]$ looks like. It classifies all points with feature value x_2 smaller than 1 as a positive point. Note that the small arrow indicates that this classifier labels all points in the direction of this arrow as belonging to the class '$+1$'. In this plot, the hypothesis classifies all points correctly except for points D and F.



Now that we have our set of hypothesis (base learners), we can proceed with implementing AdaBoost. Note throughout, we will implement the normalized version of AdaBoost, and you should round up to two decimal places for simplicity.

(a) Find h_1^* that minimizes the weighted misclassification error $\epsilon_1 = \sum_{y_i \neq h_1(x_i)} w_i^{(1)}$. To do this, create a table with two columns titled 'hypothesis' and 'ϵ_1' and one row for each hyothesis. Further, find the corresponding weight α_1. Also, find the new weights $w_i^{(2)}$ Plot your Adaboost classifier C_1 and the data (with new weights). Note that if any two hypotheses are tied, pick the one that is listed first in the definition of $\mathcal{H}$.

(b) Perform one more round of AdaBoost, report the same quantities and provide a plot of C_2. To create the plot, figure out what C_2 predicts on each of the 6 points. Then draw the classifier accordingly.

(c) Perform one final round of AdaBoost, report the same quantities and provide a plot of C_3, except that you do not need to re-calculate the weights this time.

(d) Comment on the plot of C_3. What do you notice?