# AI 391L - FALL 2025

## Homework 2: Theory

## Question 1

The perceptron is order-dependent: with the same dataset and initialization, different example orders can give different mistake counts and different final weights.

**Setup.** Predict $\text{sign}(w^\top x)$. On a mistake update $w \leftarrow w + yx$. Initialize $w = (1, 0)$. Dataset:

$$x_1 = (-1, 1),\ y_1 = +1; \quad x_2 = (1, -2),\ y_2 = +1.$$

This set is linearly separable; for example, $(w^*)^\top = (-3, -2)$ satisfies $(w^*)^\top x_1 = 1 > 0$ and $(w^*)^\top x_2 = 1 > 0$.

**Order A:** $S = (x_1, x_2)$

$$\begin{aligned} &w^{(0)} = (1, 0). \\ &x_1 : w^{(0)\top} x_1 = -1 \ \Rightarrow\ \text{mistake},\ w^{(1)} = (0, 1). \\ &x_2 : w^{(1)\top} x_2 = -2 \ \Rightarrow\ \text{mistake},\ w^{(2)} = (1, -1). \end{aligned}$$

So 2 mistakes, $w^A_{\text{final}} = (1, -1)$.

**Order B:** $S' = (x_2, x_1)$

$$\begin{aligned} &w^{(0)} = (1, 0). \\ &x_2 : w^{(0)\top} x_2 = 1 \ \Rightarrow\ \text{correct},\ w^{(1)} = (1, 0). \\ &x_1 : w^{(1)\top} x_1 = -1 \ \Rightarrow\ \text{mistake},\ w^{(2)} = (0, 1). \end{aligned}$$

So, 1 mistake, $w^B_{\text{final}} = (0, 1)$.

**Conclusion.** $S$ and $S'$ give different mistake counts (2 vs. 1) and different final weights $w^A_{\text{final}} \neq w^B_{\text{final}}$. Therefore, order matters.

## Question 2

We compare the stochastic gradient descent on the loss with the perceptron rule.

$$\phi(z) = \max(0, -z), \quad z = y\, w^\top x$$

**Step 1. Gradient**

$$\nabla_w \phi(y\, w^\top x) = \begin{cases} 0, & y\, w^\top x > 0, \\ -\, y\, x, & y\, w^\top x \le 0. \end{cases}$$

**Step 2. SGD update**

$$w_{\text{new}} = w_{\text{old}} - \eta\, \nabla_w \phi(y\, w^\top x) = \begin{cases} w_{\text{old}}, & y\, w^\top x > 0, \\ w_{\text{old}} + \eta\, y\, x, & y\, w^\top x \le 0. \end{cases}$$

**Conclusion.** This matches the perceptron algorithm with the learning rate $\eta$: - If the current example is correctly classified ($yw^\top x > 0$), no update. - If it is misclassified or on the boundary ($yw^\top x \le 0$), update $w \leftarrow w + \eta y x$.

So ... SGD on $\frac{1}{m}\sum_i \phi(y_i w^\top x_i)$ with $\phi(z) = \max(0, -z)$ is exactly the perceptron update rule.

## Question 3

**(a) Stump with error at most 1/3.** The target $c$ is a 3-piece classifier:

$$c(x) = \begin{cases} b & \theta_1 \le x \le \theta_2, \\ -b & \text{otherwise.} \end{cases}$$

$$p_L = \Pr(x < \theta_1), \quad p_M = \Pr(\theta_1 \le x \le \theta_2), \quad p_R = \Pr(x > \theta_2).$$

Clearly $p_L + p_M + p_R = 1$.

Check three simple stumps:

- Constant $-b$: error $= p_M$.
- Threshold at $\theta_1$ with left $-b$: error $= p_R$.
- Threshold at $\theta_2$ with left $b$: error $= p_L$.

So the best stump has error $\min\{p_L, p_M, p_R\} \le 1/3$.

**(b) How to find the best stump (ERM).** Given training set $\{(x_i, y_i)\}_{i=1}^m$:

1. Sort all $x_i$ (cost $O(m \log m)$).
2. For each possible threshold between two points, compute mistakes for both label directions.
3. Keep track of the threshold and polarity with the fewest mistakes.

This takes $O(m \log m)$ overall.

**(c) Why generalization works.** Decision stumps are a very simple class (VC-dim = 1). This means with enough samples $m$, the training error is close to the true error for every stump.

So the ERM stump has nearly the same true error as the best stump. From part (a) the best stump has error $\le 1/3$, so with large enough $m$ the ERM hypothesis also has error close to $\le 1/3$. Thus we can weakly learn the 3-piece class using decision stumps.

## Question 4

Show that $h_t$ has accuracy $1/2$ with respect to $D_{t+1}$.

- Define the weighted error of $h_t$ on $D_t$:

$$\varepsilon_t = \Pr_{i \sim D_t}[h_t(x_i) \neq y_i].$$

- AdaBoost sets

$$\alpha_t = \tfrac{1}{2} \ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right),$$

  and updates the distribution:

$$D_{t+1}(i) = \frac{D_t(i)\, \exp(-\alpha_t y_i h_t(x_i))}{Z_t}.$$

- Effect of the update:
  - If $h_t(x_i) = y_i$ (correct), weight is multiplied by $e^{-\alpha_t}$.
  - If $h_t(x_i) \neq y_i$ (wrong), weight is multiplied by $e^{+\alpha_t}$.
- Error of $h_t$ under $D_{t+1}$:

$$\varepsilon_t' = \sum_{\text{wrong}} D_{t+1}(i) = \frac{\varepsilon_t e^{\alpha_t}}{Z_t}.$$

- Accuracy of $h_t$ under $D_{t+1}$:

$$a_t' = \sum_{\text{correct}} D_{t+1}(i) = \frac{(1-\varepsilon_t)e^{-\alpha_t}}{Z_t}.$$

- Now compute with chosen $\alpha_t$:

$$e^{\alpha_t} = \sqrt{\frac{1-\varepsilon_t}{\varepsilon_t}}, \qquad e^{-\alpha_t} = \sqrt{\frac{\varepsilon_t}{1-\varepsilon_t}}.$$

- So both numerators equal $\sqrt{\varepsilon_t(1-\varepsilon_t)}$.
- The normalizing constant is

$$Z_t = (1-\varepsilon_t)e^{-\alpha_t} + \varepsilon_t e^{\alpha_t} = 2\sqrt{\varepsilon_t(1-\varepsilon_t)}.$$

- Therefore

$$a_t' = \varepsilon_t' = \frac{1}{2}.$$

**Conclusion:** under $D_{t+1}$, the classifier $h_t$ has accuracy exactly $1/2$.