1a) So for any x ..

$$f(x)g(x) \begin{cases} +1, & f(x) = g(x) \\ -1, & f(x) \neq g(x) \end{cases}$$

So we can write

$$1\{f(x) \neq g(x)\} = \frac{1 - f(x)g(x)}{2}$$

W.r.t. x is sampled according to D ($x \sim D$)

$$\underset{x\sim D}{\mathbb{P}}[f(x) \neq g(x)] = \underset{x\sim D}{\mathbb{E}}[1 f(x) \neq g(x)] = \underset{x\sim 1}{\mathbb{E}}\left[\frac{1 - f(x)g(x)}{2}\right] = \frac{1 - \mathbb{E}_{x\sim D}[f(x)g(x)]}{2}$$

or written as $\underset{x\sim D}{\mathbb{E}}[f(x)g(x)] = 1 - 2\underset{x\sim D}{\mathbb{P}}[f(x) \neq g(x)]$

1b) Yes! The statement holds for any domain $\mathcal{X}$ and any D on $\mathcal{X}$ provided $f, g : \mathcal{X} \to \{-1, 1\}$. Reason the identity which depends only on the range $\{-1, 1\}$ not on the domain.

---

2) So to build one polynomial tree per leaf. Lets start with indicator for variable test.

▷ if branch checks $x_i = 1$, use

$$\frac{1 + x_i}{2}$$ so that it equals $= 1$ when $x_i = 1$

$= 0$ when $x_i = -1$

▷ if branch checks $x_i = -1$, use

$$\frac{1 - x_i}{2}$$ So that it equals $= 1$ when $x_i = -1$

$= 0$ when $x_i = 1$

So for Indicator path we would get:

$$I_{path}(x) = \prod_{j=1}^{k} \frac{1 + a_j x_{ij}}{2} \longrightarrow$$

- a_j is required value (± 1) for x_{ij}
  - ↳ if all conditions are met every factor is 1, so the product is 1
  - ↳ If even one condition fails, one factor becomes 0, so the product is 0.
  - → $\boxed{\text{On/off Switch.}}$

Lets attach leaf values $b_\ell \in \{-1, 1\}$

$$P_\ell(x) = b_\ell \cdot I_{path_\ell}(x)$$

So this polynomial outputs the leaf's value *if on path* or else 0.

Finally, Sum over all t leaves:

$$p(x) = \sum_{\ell=1}^{t} P_\ell(x).$$

At any input x, exactly one path indicator equals to 1.
↳ all others are 0. So therefore Sum reduces to $\underline{\underline{p(x) = f(x)}}$

3) Totals : $\boxed{P_+: 165,}$ $\boxed{N_- = 85,}$ $\boxed{T_\# = 250}$

Lets start with Root Node. For $F \in \{X, Y, Z\}$, group them by $F=0$, $F=1$.

Compute $a = \frac{\# Pos}{\# Total}$ then $C(F=v) = 2a(1-a)$. Split Score is weighted sum by group size.

$C(D_r[Pos]) = 2 \cdot \frac{165}{250} \cdot \frac{85}{250} = \underline{0.4488}$

① Root Split

→ Split on X

- $X=0$: Pos = 105, neg = 45 → $a = {}^{105}/_{150}$, $C = 0.42$
- $X=1$: Pos = 60, neg = 40 → $a = {}^{60}/_{100}$, $C = 0.48$ weighted: 0.444

→ Split on Y

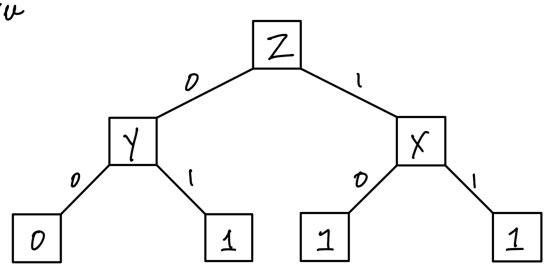
- $Y=0$: Pos = 70, neg = 50 → $a = {}^{70}/_{120}$, $C = 0.467$
- $Y=1$: Pos = 95, neg = 35 → $a = {}^{95}/_{130}$, $C = 0.391$ weighted: 0.438

→ Split on Z

- $Z=0$: Pos = 60, neg = 60 → $a = {}^{60}/_{120}$, $C = 0.50$
- $Z=1$: Pos = 105, neg = 25 → $a = {}^{105}/_{130}$, $C \approx 0.311$ weighted: 0.402

Smallest Score → root split on Z

Tree



② Depth two Split

Left child $Z=0$ (Pos: 60, Neg: 60, N: 120)

→ Split on X

- $X=0$ (45, 35) $C = 0.459$
- $X=1$ (15, 25) $C = 0.469$ $W = 0.484$

→ Split on Y

- $Y=0$ (15, 35) $C = 0.420$
- $Y=1$ (45, 25) $C = 0.459$ $W = 0.443$

Smallest Score → split on Y

→ Leaves:

$Z=0$, $Y=0$: (15, 35) → P_{neg}

$Z=0$, $Y=1$: (45, 25) → P_{pos}

Right Child: $Z=1$ (Pos: 105, Neg: 25, N = 130)

→ Split on X

- $X=0$ (60, 10) → $C = 0.245$
- $X=1$ (45, 15) → $C = 0.278$ $W = 0.305$

→ Split on Y

- $Y=0$ (55, 15) → $C = 0.321$
- $Y=1$ (50, 10) → $C = 0.278$ $W = 0.310$

Smallest Score → Split on X

→ Leaves:

$Z=1$ $X=0$ (60, 10) → P_{pos}

$Z=1$ $X=1$ (45, 15) → P_{pos}

③ Final

- Root Split on Z
  - ↳ $Z=0$: Split on Y
    - ↳ $Y=0$ → Neg
    - $Y=1$ → Pos
  - ↳ $Z=1$: split on X
    - ↳ $X=0$ → Pos
    - $X=1$ → Pos

④ Training accuracy

- $Z=0$ $Y=0$ 35
- $Z=0$ $Y=1$ 45
- $Z=1$ $X=0$ 60
- $Z=1$ $X=1$ 45

$T_c = 185$

$Acc = {}^{185}/_{250} = 0.74$ or $\underline{\underline{74\%}}$

4) Pac Learning Algorithm

Training set $S = \{x_1, y_1, \ldots, x_m, y_m\}$ where $x_i \in R$ $y_i \in \{-1, 1\}$

① Input: m labeled point $S = \{(x_i, y_i)\}$ with $y_i \in \{-1, +1\}$

② set $\hat{\theta}$ = "no value" # $\hat{\theta}$ - learned threshold.

③ Scan the sorted list from L or R:

if $y_j = -1$, set $\hat{\theta} = x_j$ # Keep largest x with label -1

④ if $\hat{\theta}$ has no value (there was no -1 in S)

set $\hat{\theta} = x_1 - 1$ # makes all points predict 1

⑤ output classifier h_{θ}:

$h_{\theta}x = -1$ if $x \leq \theta$, else $+1$

Time: sort $O(m \log m)$, scan $O(m)$

Assume there is some true threshold that labels the data correctly. Our algorithm chooses the learned threshold as the largest sample with label -1. Only mistakes can happen between learned and true threshold. If we get at least one training point close enough to true threshold, then total error is at most ε. The chance that no point lands in that region is at most $(1-\varepsilon)^m$, which is smaller than $e^{-\varepsilon m}$. To make this failure probability at most δ it is enough to take

$$m \geq \frac{1}{\varepsilon} \ln \frac{1}{\delta}$$

So with probability at least $1-\delta$, the algorithm finds a classifier whose error is at most ε.

5a) if $err(h') > \varepsilon$, each fresh example is correct with prob $< 1 - \varepsilon$. For a block of size k:

$$Pr[h' \text{ makes 0 mistakes on the block}] = (1-\varepsilon)^k \leq e^{-\varepsilon k}$$

Pick $k > \frac{1}{\varepsilon} \ln \frac{1}{\delta'}$. Then

$$Pr[err(h') > \varepsilon \text{ and } h' \text{ passes its block}] \leq \delta'$$

5b) Learner A makes at most t mistakes, so it visits at most $t+1$ hypotheses $h_1, \ldots, h_{t+1}$.

Use union bound with $\delta' = \frac{\delta}{(t+1)}$

Choose

$$k \geq \frac{1}{\varepsilon} \ln \frac{t+1}{\delta}$$

5c) PAC Learner B (uses A as subroutine)

① Compute block size k as above.

② Draw $m = (t+1) \cdot k$ training examples.

③ Split them into $t+1$ blocks, each of size k.

④ Start with A's first hypothesis h_1.

⑤ For each block $j = 1, \ldots, t+1$:

- Test hypothesis h_j on the block j.
- if it makes no mistakes, output h_j and stop.
- if it makes a mistake, feed one mistake back to A so it updates to h_{j+1}.

⑥ if none pass, output the last hypothesis.

NOTE

- Each bad hypothesis passes its block with probability $\leq \delta/(t+1)$
- with at most $t+1$ hypotheses, union bound gives failure probability $\leq \delta$
- So with probability $\geq 1-\delta$, the output has error $\leq \varepsilon$.

Samples Used:

$$m = (t+1) \cdot k = \frac{t+1}{\varepsilon} \ln \frac{t+1}{\delta}.$$