ONLINE LEARNING

Learning as the data comes in; no sample from which to generate a hypothesis. The learner gets feedback & must respond adaptively.

Setting Time $t = 1, 2, 3, \ldots$ at time t

1. The learner is given an example x_t
2. The learner must predict a label $\hat{y}_t$
3. Then the true label y_t is revealed
4. The learner updates its prediction rule based on x_t, y_t & $\hat{y}_t$.

No assumption is made that x_t are drawn i.i.d from some sample space. In fact no assumptions about distributions at all. The analysis is adversarial: we assume an adversary who chooses x_t to try & force the learner to make as many mistakes as possible.

The learner makes a mistake in round t if $y_t \neq \hat{y}_t$.

How do we judge performance? It can't be just # of mistakes since the adversary can force a mistake at any time. We define regret

After receiving $x_1, x_2 \ldots x_t, \ldots x_T$

R_T = number of mistakes − number of mistakes made by the best predictor from some given class of predictors.

Realizable setting

We assume a class of hypotheses $\mathcal{H}$ s.t. the sequence $(x_1, y_1) \cdots (x_t, y_t) \cdots (x_T, y_T)$ is consistent with some $h \in \mathcal{H}$. In this setting the best ~~compar~~ predictor makes 0 mistakes so we aim to given an upper bound on the number of mistakes.

The learner knows $\mathcal{H}$ & the adversary <u>chooses</u> some $h^* \in \mathcal{H}$

Given $S = ((x_1, h^*(x_1)), \ldots (x_T, h^*(x_T)))$

Suppose A is an (adaptive) algorithm to make predictions we define for S

$M_A(S)$: # of mistakes made by A on S.

We write $M_A(\mathcal{H})$ for sup of $M_A(S)$ for all choices of S & h^*. If we can prove $M_A(\mathcal{H}) \le B < \infty$ we have a mistake bound & $\mathcal{H}$ is online learnable.

We assume $\mathcal{H}$ finite

<u>Basic Alg</u> input $\mathcal{H}$

initialize $V_1 = \mathcal{H}$

for $t = 1 \ldots T$ do

read $\vec{x}_t$

choose $h \in V_t$ & predict $h(\vec{x}_t)$

receive y_t (* which is $h^*(\vec{x}_t)$ *)

update $V_{t+1} = \{h \in V_t \mid h(x_t) = y_t\}$

Easy to see $M(\mathcal{H}) \le |\mathcal{H}| - 1$ This is no good.

<u>Halving algorithm</u> : init as above

for $t = 1 .. T$ do

1. read x_t

VOTE! 2. predict $\hat{y}_t = \underset{r \in \{0,1\}}{\operatorname{argmax}} \left|\{h \in V_t \mid h(\vec{x}_t) = r\}\right|$

3. receive true label $y_t = h^*(x_t)$

4. update $V_{t+1} = \{h \in V_t \mid h(x_t) = y_t\}$ END.

If the alg makes a mistake, at least 1/2 of the hypotheses are rejected so

$$1 \le |V_{T+1}| \le 2^{-M} |\mathcal{H}|$$

so $M \le \log_2 |\mathcal{H}|$

<u>Example</u> Instances are bit sequences

$$\mathcal{H} = \{ \text{ends in } 0, \; \text{ends in } 00, \; \ldots\ldots, \; \text{ends in } 0^k \}$$

Sequence $10, 100, 1000, \ldots, 1\underbrace{00\cdots0}_{k}$

suppose h^* is ends in 0^k.

~~Then~~ Suppose you pick the rule with smallest # of zeros. Then only at the end do you finally hit the right rule so $k-1$ mistakes.

Halving would never make more than $\log_2 k$ mistakes. In this example it would make at most 1 mistake.

—x—

<u>Prop</u> Given $\mathcal{H}$ with VC dimension d, for <u>any</u> <u>deterministic</u> <u>online</u> algorithm, there is a sequence of inputs for which the algorithm makes at least d mistakes.

<u>Proof</u> We assume the adversary knows your strategy & it chooses the label after the learner does so it can look at what the algorithm has chosen and pick a different one, of course it has to be consistent with something in $\mathcal{H}$. Now there is some set of d points which is shattered by $\mathcal{H}$. The adversary chooses these as the first d instances. Since any labelling of these d points is achievable, adversary ~~we~~ can ensure that the algorithm mislabels all of them.

<u>Remark</u> Even with a randomized algorithm one can show that the adversary can force at least $d/2$ errors in expectation.

A general lowerbound for online learning:

The adversary needs a scheme to ensure an algorithm makes as many mistakes as possible. The adversary strategy can be described as a tree as follows. The tree is a complete binary tree and at every node we have an instance of X. The depth of the tree is T. As the algorithm proceeds the adversary goes down the tree. At each round the learner is given the instance of ~~x~~ X at that node. If the learner predicts 0 the adversary says "that's wrong!" & goes right. If the learner says 1, the adversary says "that's wrong" and goes left. Then if the true label is 1, go right if the true label is 0 go left.

The adversary wins if the learner is wrong at every step. The tree represents a winning strategy for Adversary only if $\forall (y_1, \cdots y_T) \in \{0,1\}^T$ $\exists h \in \mathcal{H}$ s.t. $h(x_t) = y_t$ where x_t is the item in the tree reached by following the path $y_1 \cdots y_{t-1}$.

Def An $\mathcal{H}$-shattered tree of depth d is a sequence of instances $x_1, x_2, \ldots\ x_{2^{d+1}-1}$ in X s.t. $\forall (y_1, \cdots y_d) \in \{0,1\}^d$ $\exists h \in \mathcal{H}$ s.t. $h(x_{i_t}) = y_t$ where

i_t is the node obtained by following $y_1 \cdots y_{t-1}$.

Note $\quad i_{t+1} = 2 i_t + y_t$ or $i_t = 2^{t-1} + \sum_{j=1}^{t-1} y_j 2^{t-1-j}$

So we are not achieving every labelling of the elements but every labelling gives some distinct path.

Def $L\dim(\mathcal{H})$ is the largest integer d s.t. $\exists$ an $\mathcal{H}$-shattered tree of depth d.

Lemma No algorithm can have a mistake bound less than $L\dim(\mathcal{H})$.

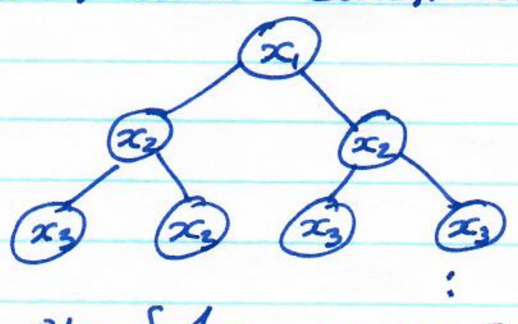
Proof Immediate from the definition.

Ex 1 $\mathcal{H}$ finite $L\dim(\mathcal{H}) \leq \log_2(\mathcal{H})$

Ex 2 $X = \{1, \ldots n\}$ $\mathcal{H} = \{h_1, \cdots h_n\}$ $h_j(x) = 1$ iff $x = j$
You can only make the trivial tree no matter how big n is so $L\dim = 1$. For the learner it is easy: keep predicting 0, the first time you make a mistake you know h.

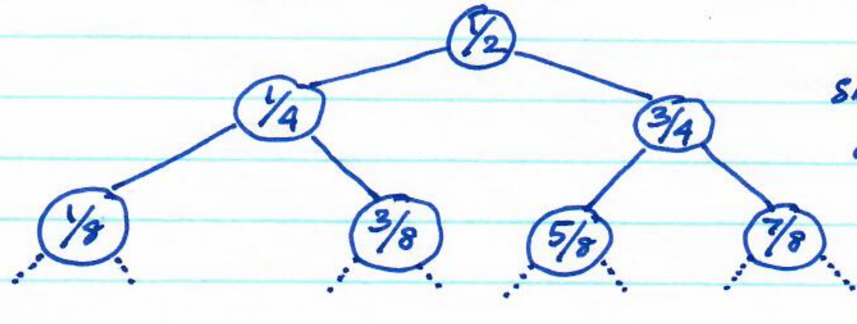
~~Ex3~~ Prop VC dim $\leq$ L dim

Proof Suppose VC dim $= d$ so $\{x_1, \cdots x_d\}$ can be shattered. Then construct the tree



Clearly this is a shattered tree since any path can be shattered.

Ex3 $X = [0,1]$ $\mathcal{H} = \{ 1_{[x>a]} : a \in [0,1] \}$ Threshold f^{ns}



This tree is shattered by $\mathcal{H}$ and can be of any depth

$$L\dim(\mathcal{H}) = \infty \ \& \ VC\dim(\mathcal{H}) = 1$$