CSCI 678: Theoretical Machine Learning
Lecture 6

Fall 2024, Instructor: Haipeng Luo

1 Online Binary Classification: Littlestone Dimension

In the last lecture, we started discussing online learning and derived the following sequence of upper bounding on the sequential value $\mathcal{V}^{\text{seq}}(\mathcal{F}, n)$:

$$\begin{aligned}\mathcal{V}^{\text{seq}}(\mathcal{F}, n) &\le \sup_{\mathcal{P}\in\Delta(\mathcal{Z}^n)} \mathop{\mathbb{E}}_{z_{1:n}\sim\mathcal{P}} \left[\sup_{f\in\mathcal{F}} \frac{1}{n}\sum_{t=1}^{n} \left(\mathbb{E}_{z'_t\sim\mathcal{P}(\cdot|z_{1:t-1})}\left[\ell(f, z'_t)\right] - \ell(f, z_t)\right)\right] \\ &\le 2\mathcal{R}^{\text{seq}}(\ell(\mathcal{F})) \le \mathcal{R}^{\text{seq}}(\mathcal{F}) \le \sup_{\boldsymbol{x}} \sqrt{\frac{2\ln \mathcal{N}_0(\mathcal{F}|_{\boldsymbol{x}})}{n}},\end{aligned}$$

where the first two steps hold generally and the last two are specific for binary classification with 0-1 loss. Similarly to previous discussions for statistical learning, we will now introduce a combinatorial parameter that provides a reasonable upper bound for $\mathcal{N}_0(\mathcal{F}|_{\boldsymbol{x}})$ and that is easier to bound.

First, we say that $\mathcal{F}$ *shatters a tree* $\boldsymbol{x}$ if for any $\boldsymbol{\epsilon} \in \{-1, +1\}^n$, there exists $f \in \mathcal{F}$ such that $f(\boldsymbol{x}_t(\boldsymbol{\epsilon})) = \epsilon_t$ holds for all $t = 1, \dots, n$. Then, we define the *Littlestone dimension* $\text{Ldim}(\mathcal{F})$ of a class $\mathcal{F}$ as the depth of the largest tree that can be shattered by $\mathcal{F}$ ($\text{Ldim}(\mathcal{F})$ is defined as 0 if no tree can be shattered by $\mathcal{F}$, and ∞ if for any n there exists a tree of depth n that is shattered by $\mathcal{F}$).[1]

Note that one always has $\text{VCdim}(\mathcal{F}) \le \text{Ldim}(\mathcal{F})$, since if $x_{1:n}$ is shattered by $\mathcal{F}$, then a tree $\boldsymbol{x}$ of depth n such that all nodes at level t have value x_t for $t = 1, \dots, n$ is clearly shattered by $\mathcal{F}$ as well. Using this fact, one sees that $\text{Ldim}(\mathcal{F}) = 0$ implies $\text{VCdim}(\mathcal{F}) = 0$ and thus $\mathcal{F}$ contains only one function.

Similar to VC dimension, to prove $\text{Ldim}(\mathcal{F}) = d$ we have to 1) construct a tree of depth d that can be shattered by $\mathcal{F}$ and 2) prove that no tree of depth $n + 1$ can be shattered by $\mathcal{F}$. As an example, we go back to the simple class discussed in the last lecture

$$\mathcal{F} = \left\{ f_\theta(x) = \begin{cases} +1, & \text{if } x = \theta \\ -1, & \text{else} \end{cases} \;\middle|\; \theta \in \mathbb{R} \right\}. \tag{1}$$

and argue that it has Littlestone dimension exactly 1 (the same as its VC dimension). This is because it clearly can shatter a tree with depth 1 (in fact, any tree with depth 1); on the other hand, it cannot shatter any tree with depth 2 (just consider $\boldsymbol{\epsilon} = (+1, +1)$ and $\boldsymbol{\epsilon} = (+1, -1)$). In HW2, you will see a generalization of this example from one dimension to general dimension.

Littlestone dimension is very useful since it can be used to provide a good upper bound on the zero-covering number, in a way very similar to the Sauer's lemma. This is stated in the following theorem, whose proof also reveals how to construct a zero-cover recursively.

[1] We point out the subtlety that in the definition of sequential Rademacher complexity and shattering, $\boldsymbol{\epsilon}$ represents both a path and a labeling, while in the definition of zero-covering, $\boldsymbol{\epsilon}$ represents a path only.

Theorem 1. *Suppose that $\boldsymbol{x}$ is any $\mathcal{X}$-valued tree with depth n and $\mathcal{F} \subset \{-1,+1\}^{\mathcal{X}}$ has Littlestone dimension $d \le n$, then*

$$\mathcal{N}_0(\mathcal{F}|_{\boldsymbol{x}}) \le \sum_{i=0}^{d} \binom{n}{i} \le \left(\frac{en}{d}\right)^d.$$

Proof. Let $g(d,n) = \sum_{i=0}^{d} \binom{n}{i}$. We will prove $\mathcal{N}_0(F|_{\boldsymbol{x}}) \le g(d,n)$ via induction on the value of $d+n$ (the second inequality $\sum_{i=0}^{d} \binom{n}{i} \le \left(\frac{en}{d}\right)^d$ has already been proven in Sauer's lemma). The base case $d+n=1$ is trivial since the only configuration is $d=0$ and $n=1$, in which case $\mathcal{N}_0(F|_{\boldsymbol{x}}) = 1$ clearly. Now, we assume that the statement holds for any $n' > d'$ such that $n'+d' < n+d$, and prove $\mathcal{N}_0(F|_{\boldsymbol{x}}) \le g(d,n)$. The case when $d=0$ is again trivial, so we assume $d>0$.

First, we provide a way to recursively construct a zero-cover for $\mathcal{F}|_{\boldsymbol{x}}$. Depending on the prediction for the root of the tree (which we denote by $\boldsymbol{x}_1$ for simplicity), we split the class $\mathcal{F}$ into two subclasses: $\mathcal{F}_- = \{f \in \mathcal{F} \mid f(\boldsymbol{x}_1) = -1\}$ and $\mathcal{F}_+ = \{f \in \mathcal{F} \mid f(\boldsymbol{x}_1) = -1\}$. Let $\boldsymbol{x}^\ell$ and $\boldsymbol{x}^r$ be the left and right subtrees of the root of $\boldsymbol{x}$ (which have depth $n-1$), and V_+^ℓ and V_+^r be the smallest zero-cover of $\mathcal{F}_+|_{\boldsymbol{x}^\ell}$ and $\mathcal{F}_+|_{\boldsymbol{x}^r}$ respectively. Now we construct a set V_+ in the following way: 1) the root of every tree in V_+ has value $+1$, 2) pair the element from V_+^ℓ and V_+^r to form the left and right subtrees of the root for trees in V_+, so that each element from V_+^ℓ and V_+^r appears at least once. It is clear that this can be done such that $|V_+| = \max\left\{|V_+^\ell|, |V_+^r|\right\}$. Moreover, it is also clear that V_+ is a zero-cover of $\mathcal{F}_+|_{\boldsymbol{x}}$. In the exact same way, we construct V_- such that $|V_-| = \max\left\{|V_-^\ell|, |V_-^r|\right\}$ and V_- is a zero-cover of $\mathcal{F}_-|_{\boldsymbol{x}}$. Finally, we have that $V = V_- \cup V_+$ is a zero-cover of $\mathcal{F}|_{\boldsymbol{x}}$.

It remains to bound $|V_-|$ and $|V_+|$. The key observation is that it is impossible that $\mathcal{F}_-$ and $\mathcal{F}_+$ both have Littlestone dimension d. Otherwise, there are trees $\boldsymbol{x}^-$ and $\boldsymbol{x}^+$ of depth d that can be shattered by $\mathcal{F}_-$ and $\mathcal{F}_+$ respectively. By pairing $\boldsymbol{x}^-$ and $\boldsymbol{x}^+$ as the left and right subtrees of the root $\boldsymbol{x}_1$, we obtain a tree with depth $d+1$ that can be shattered by $\mathcal{F}$, which is a contradiction to $\mathrm{Ldim}(\mathcal{F}) = d$. Without loss of generality, we can thus assume $\mathcal{F}_-$ has Littlestone dimension at most $d-1$. Using the inductive hypothesis, we thus have

$$|V_+| = \max\left\{|V_+^\ell|, |V_+^r|\right\} = \max\left\{\mathcal{N}_0(\mathcal{F}_+|_{\boldsymbol{x}^\ell}), \mathcal{N}_0(\mathcal{F}_+|_{\boldsymbol{x}^r})\right\} \le g(d, n-1),$$

and

$$|V_-| = \max\left\{|V_-^\ell|, |V_-^r|\right\} = \max\left\{\mathcal{N}_0(\mathcal{F}_-|_{\boldsymbol{x}^\ell}), \mathcal{N}_0(\mathcal{F}_-|_{\boldsymbol{x}^r})\right\} \le g(d-1, n-1).$$

Therefore, $\mathcal{N}_0(\mathcal{F}|_{\boldsymbol{x}}) \le |V| = |V_-| + |V_+| \le g(d-1,n-1) + g(d,n-1) = g(d,n)$, where the last equality is proven in Sauer's lemma already. This finishes the proof. □

We remark that the concept of zero-covering is the key to allow a construction with $|V_+| = \max\left\{|V_+^\ell|, |V_+^r|\right\}$. If we focus on the projection instead, we would have arrived at something like $|V_+| = |V_+^\ell| \times |V_+^r|$. Applying this theorem directly, we conclude that the zero-covering number of the simple class defined by Equation (1) is indeed bounded by $g(1,n) = n+1$ (even if the tree contains identical elements on some paths). The proof of Theorem 1 also reveals a recursive way to construct such a cover with size $n+1$ (you should try this for a small tree of depth say 3).

The following bound on the value of the game is a direct corollary based on previous discussions.

Corollary 1. *For any class of binary classifier $\mathcal{F}$ with $d = \mathrm{Ldim}(\mathcal{F})$, we have*

$$\mathcal{V}^{\mathrm{seq}}(\mathcal{F}, n) \le \mathcal{R}^{\mathrm{seq}}(\mathcal{F}) \le \sqrt{\frac{2d \ln\left(\frac{en}{d}\right)}{n}}.$$

So finite Littlestone dimension is sufficient for online learnability. It turns out that it is also *necessary* for online learnability, indicating that this sequence of upper bounding is tight in a sense.

Theorem 2. *If $\mathrm{Ldim}(\mathcal{F}) = \infty$, then $\mathcal{V}^{\mathrm{seq}}(\mathcal{F}, n) \ge 1/2$ for any n.*

Proof. Let $\boldsymbol{x}$ be a tree of depth n shattered by $\mathcal{F}$ (which always exists since $\mathrm{Ldim}(\mathcal{F}) = \infty$), and $\epsilon_1, \dots, \epsilon_n$ be i.i.d. Rademacher random variables. Consider an environment that chooses $(\boldsymbol{x}_t(\boldsymbol{\epsilon}), \epsilon_t)$ at time t. In such an environment, no matter what the algorithm is (proper or improper), its expected

total loss is always exactly $n/2$. On the other hand, by the definition of shattering, there is always an $f \in \mathcal{F}$ with perfect prediction on this dataset, which means that the expected regret is exactly $n/2$ and thus $\mathcal{V}^{\text{seq}}(\mathcal{F}, n) \geq 1/2$. □

In fact, in HW2 you will prove an even stronger statement when $\text{Ldim}(\mathcal{F}) \leq n$: $\mathcal{V}^{\text{seq}}(\mathcal{F}, n) \geq \sqrt{\frac{\text{Ldim}(\mathcal{F})}{8n}}$, further showing that the upper bound we obtain is tight.

Summary. Combining all steps, we have shown for binary classification problems:

$$\sqrt{\frac{d}{8n}} \leq \mathcal{V}^{\text{seq}}(\mathcal{F}, n) \leq \sup_{\mathcal{P} \in \Delta(\mathcal{Z}^n)} \mathop{\mathbb{E}}_{z_{1:n} \sim \mathcal{P}} \left[\sup_{f \in \mathcal{F}} \frac{1}{n} \sum_{t=1}^{n} \left(\mathbb{E}_{z_t' \sim \mathcal{P}(\cdot | z_{1:t-1})} [\ell(f, z_t')] - \ell(f, z_t) \right) \right]$$

$$\leq 2\mathcal{R}^{\text{seq}}(\ell(\mathcal{F})) \leq \mathcal{R}^{\text{seq}}(\mathcal{F}) \leq \sup_{\boldsymbol{x}} \sqrt{\frac{2 \ln \mathcal{N}_0(\mathcal{F}|_{\boldsymbol{x}})}{n}} \leq \sqrt{\frac{2d \ln\left(\frac{en}{d}\right)}{n}}$$

where $d = \text{Ldim}(\mathcal{F}) \leq n$.

1.1 Online learning is strictly harder

In Lecture 1, via the online-to-batch conversion we showed that online learning is at least as hard as statistical learning. Is it strictly harder? The example of the simple class defined by Equation (1) does not indicate that because the VC dimension and the Littlestone dimension coincide (both are 1). Instead, let's consider the threshold function class defined over $\mathcal{X} = \mathbb{R}$:

$$\mathcal{F} = \left\{ f_\theta(x) = \begin{cases} +1 & \text{if } x \leq \theta \\ -1 & \text{else} \end{cases} \;\middle|\; \theta \in \mathbb{R} \right\}, \tag{2}$$

which has VC dimension exactly 1 as discussed in Lecture 2. It turns out that the Littlestone dimension of this seemingly simple class is infinity!

Proposition 1. *The Littlestone dimension of the threshold function class (Equation (2)) is ∞.*

Proof. To see this, consider an infinite $[0, 1]$-valued tree $\boldsymbol{x}$ with root being $1/2$, and the left child and right child of a node with value a at level t being $a - \frac{1}{2^{t+1}}$ and $a + \frac{1}{2^{t+1}}$ respectively. (This is much easier to interpret if you draw a picture.)

Now for any n and any path/labeling $\boldsymbol{\epsilon}$, let θ_1 be the last node of this path $\boldsymbol{x}_n(\boldsymbol{\epsilon})$ and θ_2 be the last node on this path with label $-\epsilon_n$. Then the claim is that any value θ in between θ_1 and θ_2 satisfies: $f_\theta(\boldsymbol{x}_t(\boldsymbol{\epsilon})) = \epsilon_t$ for all $t = 1, \ldots, n$, and thus $\mathcal{F}$ shatters this tree. Indeed, note that the tree is constructed such that if $\epsilon_t = +1$, then every node in the path below level t has value larger than the node at level t. Similarly, if $\epsilon_t = -1$, then every node in the path below level t has value smaller than the node at level t. Therefore, suppose $\epsilon_n = +1$, then all the nodes with label $+1$ on the path must have a value smaller than θ_1, and all the nodes with label -1 on the path must have a value larger than θ_2, and thus f_θ predicts all the labels correctly if $\theta \in [\theta_1, \theta_2)$. The case for $\epsilon_n = -1$ is similar. □

Based on previous discussions, we conclude that this simple class is learnable in the statistical learning setting, but not learnable in the online setting. More generally, it is clear that the class of linear classifiers

$$\mathcal{F} = \left\{ f_{\theta,b}(x) = \text{sign}\left(\langle x, \theta \rangle + b\right) \mid \theta \in \mathbb{R}^d, b \in \mathbb{R} \right\}$$

also has infinite Littlestone dimension (since it subsumes the threshold class), while having a finite VC dimension $d + 1$. This illustrates that online learning is not just as hard as statistical learning, but is in fact *strictly* harder than statistical learning.

However, is online learning just too hard to be meaningful, if even learning linear classifiers is impossible? The answer is yes in some sense for online classification, or really, for the 0-1 loss. In fact, even though 0-1 loss is seemingly not as challenging for statistical learning since any class with finite VC dimension is learnable using ERM, note that implementing ERM could still be *computationally* hard even for the class of linear classifiers. Therefore, in practice, we rarely directly learn

with 0-1 loss anyway, whether in the statistical setting or the online setting. Instead, we often use some nice surrogate losses, making the problem closer to a regression problem. Indeed, as we will see starting from the next section, many more possibilities open up for online regression.

2 Online Regression

To study online regression with a function class $\mathcal{F} \subset \mathcal{Y}^{\mathcal{X}}$ for $\mathcal{Y} = [-1, +1]$, the first key concept is still covering. We say that a set V of $[-1, +1]$-valued trees is an *α-cover* (with respect to ℓ_p norm) of $\mathcal{F}|_{\boldsymbol{x}}$ for a $\mathcal{X}$-valued tree $\boldsymbol{x}$ if

$$\forall f \in \mathcal{F}, \forall \boldsymbol{\epsilon} \in \{-1, +1\}^n, \exists \boldsymbol{v} \in V, \text{s.t. } \left(\frac{1}{n} \sum_{t=1}^{n} |f(\boldsymbol{x}_t(\boldsymbol{\epsilon})) - \boldsymbol{v}_t(\boldsymbol{\epsilon})|^p \right)^{1/p} \leq \alpha.$$

Note that the order of the quantifiers $\forall \boldsymbol{\epsilon}$ and $\exists v$ is consistent with the zero-covering definition for online classification, and also that this serves as an exact analogue of α-cover for the statistical learning setting. The α-covering number $\mathcal{N}_p(\mathcal{F}|_{\boldsymbol{x}}, \alpha)$ is simply the size of the minimum α-cover (with respect to ℓ_p norm). As always, by definition $\mathcal{N}_p(\mathcal{F}|_{\boldsymbol{x}}, \alpha)$ is non-increasing in α, and the normalization is such that $\mathcal{N}_p(\mathcal{F}|_{\boldsymbol{x}}, \alpha)$ is non-decreasing in p.

An α-cover naturally allows us to move from an infinite class to a finite class and derive the following bound on the sequential Rademacher complexity:

Theorem 3. *For any $\mathcal{X}$-valued tree $\boldsymbol{x}$ and $\mathcal{F} \subset [-1, +1]^{\mathcal{X}}$, we have*

$$\widehat{\mathcal{R}}^{\text{seq}}(\mathcal{F}; \boldsymbol{x}) \leq \inf_{\alpha \geq 0} \left(\alpha + \sqrt{\frac{2 \ln \mathcal{N}_1(\mathcal{F}|_{\boldsymbol{x}}, \alpha)}{n}} \right).$$

Proof. Fix any α. Let V be an α-cover of $\mathcal{F}|_{\boldsymbol{x}}$ with size $\mathcal{N}_1(\mathcal{F}|_{\boldsymbol{x}}, \alpha)$ and $\boldsymbol{v}^{f,\boldsymbol{\epsilon}} \in V$ be the tree that "certifies" the cover for a given f and $\boldsymbol{\epsilon}$. We then have

$$\begin{aligned}
\widehat{\mathcal{R}}^{\text{seq}}(\mathcal{F}; \boldsymbol{x}) &= \frac{1}{n} \mathbb{E} \left[\sup_{f \in \mathcal{F}} \sum_{t=1}^{n} \epsilon_t f(\boldsymbol{x}_t(\boldsymbol{\epsilon})) \right] \\
&= \frac{1}{n} \mathbb{E} \left[\sup_{f \in \mathcal{F}} \sum_{t=1}^{n} \epsilon_t (f(\boldsymbol{x}_t(\boldsymbol{\epsilon})) - \boldsymbol{v}_t^{f,\boldsymbol{\epsilon}}(\boldsymbol{\epsilon}) + \boldsymbol{v}_t^{f,\boldsymbol{\epsilon}}(\boldsymbol{\epsilon})) \right] \\
&\leq \frac{1}{n} \mathbb{E} \left[\sup_{f \in \mathcal{F}} \sum_{t=1}^{n} \epsilon_t (f(\boldsymbol{x}_t(\boldsymbol{\epsilon})) - \boldsymbol{v}_t^{f,\boldsymbol{\epsilon}}(\boldsymbol{\epsilon})) \right] + \frac{1}{n} \mathbb{E} \left[\sup_{f \in \mathcal{F}} \sum_{t=1}^{n} \epsilon_t \boldsymbol{v}_t^{f,\boldsymbol{\epsilon}}(\boldsymbol{\epsilon}) \right] \\
&\leq \alpha + \frac{1}{n} \mathbb{E} \left[\sup_{v \in V} \sum_{t=1}^{n} \epsilon_t \boldsymbol{v}(\boldsymbol{\epsilon}) \right] \leq \alpha + \sqrt{\frac{2 \ln \mathcal{N}_1(F|_{\boldsymbol{x}}, \alpha)}{n}},
\end{aligned}$$

where the last step uses the finite class bound (Theorem 3 of Lecture 5). □

Using similar chaining techniques, one can also tighten the bound via Dudley entropy integral (proof omitted):

Theorem 4. *For any $\mathcal{X}$-valued tree $\boldsymbol{x}$ and $\mathcal{F} \subset [-1, +1]^{\mathcal{X}}$, we have*

$$\widehat{\mathcal{R}}^{\text{seq}}(\mathcal{F}; \boldsymbol{x}) \leq \inf_{0 \leq \alpha \leq 1} \left(4\alpha + \frac{12}{\sqrt{n}} \int_{\alpha}^{1} \sqrt{\ln \mathcal{N}_2(\mathcal{F}|_{\boldsymbol{x}}, \delta)} d\delta \right).$$

At this point, it is probably not surprising that there is again a combinatorial parameter that provides a nice upper bound on the covering number. To introduce such a parameter, we first generalize the shattering concept: a tree $\boldsymbol{x}$ is α-shattered by a class $\mathcal{F} \in [-1, +1]^{\mathcal{X}}$ if there exists a $[-1, +1]$-valued tree $\boldsymbol{y}$ (called the witness) such that

$$\forall \boldsymbol{\epsilon} \in \{-1, +1\}^n, \exists f \in \mathcal{F}, \text{s.t. } \epsilon_t (f(\boldsymbol{x}_t(\boldsymbol{\epsilon})) - \boldsymbol{y}_t(\boldsymbol{\epsilon})) \geq \alpha/2 \text{ holds for all } t = 1, \ldots, n.$$

The (sequential) fat-shattering dimension of $\mathcal{F}$ at scale α, denoted by $\text{sfat}(\mathcal{F}, \alpha)$, is then defined as the depth of the largest tree that can be α-shattered by $\mathcal{F}$. The following theorem shows the connection between covering number and fat-shattering dimension (proof omitted):

Theorem 5. *For any $\alpha > 0$, tree $\boldsymbol{x}$ of depth n, and function class $\mathcal{F} \subset [-1, +1]^{\mathcal{X}}$, we have* $\ln \mathcal{N}_\infty(\mathcal{F}|_{\boldsymbol{x}}, \alpha) \leq \text{sfat}(\mathcal{F}, \alpha) \ln \left(\frac{2en}{\alpha} \right)$.

Example 1: linear functions Recall the linear class defined by $\mathcal{F} = \left\{f_\theta(x) = \langle \theta, x\rangle \mid \theta \in B_p^d\right\}$ and $\mathcal{X} = B_q^d$ for some $p \geq 1$ and $q \geq 1$ such that $1/p + 1/q = 1$. We have shown that this class admits a *pointwise* α-cover $\mathcal{H}$ of size $\left(\frac{2}{\alpha} + 1\right)^d$ (Proposition 2 of Lecture 3). It is clear that the projection $\mathcal{H}|_{\boldsymbol{x}}$ of this pointwise cover for a tree $\boldsymbol{x}$ is an α-cover for $\mathcal{F}|_{\boldsymbol{x}}$, and thus $\mathcal{N}_\infty(\mathcal{F}|_{\boldsymbol{x}}, \alpha) \leq \left(\frac{2}{\alpha} + 1\right)^d$. According to previous calculations, the Dudley entropy integral bound gives $\mathcal{R}^{\text{seq}}(\mathcal{F}) \leq \mathcal{O}(\sqrt{d/n})$. So linear class is online learnable, at the same rate as in the statistical learning setting.

Example 2: non-decreasing functions We have shown that in the statistical learning setting the class of all non-decreasing $[-1, +1]$-valued functions defined over $\mathbb{R}$ has α-covering number $(n+1)^{1/\alpha}$ and fat-shattering dimension $4/\alpha$. What about the online setting? We argue that no meaningful covering number or fat-shattering dimension can be derived, since the problem is simply not online learnable. To see this, first recall the tree we constructed in the proof of Proposition 1: an infinite $[0, 1]$-valued tree $\boldsymbol{x}$ with root being $1/2$, and the left child and right child of a node with value a at level t being $a - \frac{1}{2^{t+1}}$ and $a + \frac{1}{2^{t+1}}$ respectively.

The environment is almost the same as that in the proof of Theorem 2: at time t, picks $(x_t, y_t) = (\boldsymbol{x}_t(-\boldsymbol{\epsilon}), \epsilon_t)$ where $\epsilon_1, \dots, \epsilon_n$ are i.i.d. Rademacher random variables (note the opposite sign of the path compared to Theorem 2). Let the loss of the problem be the square loss $\ell(f, (x, y)) = (f(x) - y)^2$. Then no matter how the algorithm behaves (properly or improperly), because y_t is $+1$ or -1 with equal probability, the square loss of the learner is always at least $\frac{1}{2}\min_{\widehat{y}}\left((\widehat{y} - 1)^2 + (\widehat{y} + 1)^2\right) = 1$. On the other hand, by the construction of the tree, we always have $x_t \leq x_{t'}$ for any t and t' such that $y_t = -1$ and $y_{t'} = +1$, and therefore we can always find a non-decreasing function that passes all the points and has zero square loss. This makes the learner suffer linear regret.

We remark without going into details that, nevertheless, the problem becomes learnable in the so-called "transductive" setting, where the set of $\{x_1, \dots, x_T\}$ is known to the learner ahead of the time (but not the order of $x_{1:T}$ nor the outcomes $y_{1:T}$ of course).

Bridging classification and regression. We briefly discuss how the discussion on regression here is useful for classification, as hinted at the end of the last section. First, note that most binary classifier class takes the form $\mathcal{H} = \{\text{sign}(f(\cdot)) \mid f \in \mathcal{F}\}$ for some real-valued class $\mathcal{F}$, where each $f \in \mathcal{F}$ outputs some "score", so that the sign of the score predicts the label and the magnitude of the score represents the "confidence" of the prediction. In this case, there is no difference is picking $\mathcal{F}$ or $\mathcal{H}$ as the decision space and reference class for our problem. If we pick $\mathcal{F}$ instead, then the 0-1 loss becomes $\ell(f, (x, y)) = \mathbf{1}\{yf(x) \leq 0\}$ and the problem involves dealing with a real-valued function class. The problem is of course still not online learnable if $\text{Ldim}(\mathcal{H}) = \infty$ since all we have done is to represent the same problem slightly differently. However, we could now choose another loss function $\ell'(f, (x, y))$ that is an upper bound of the 0-1 loss and is online learnable as the surrogate for optimizing 0-1 loss. Clearly, we then have

$$\sum_{t=1}^T \mathbf{1}\{y_t f_t(x_t) \leq 0\} \leq \sum_{t=1}^T \ell'(f_t, (x_t, y_t)) = \inf_{f \in \mathcal{F}} \sum_{t=1}^T \ell'(f, (x_t, y_t)) + \text{Reg}(\mathcal{F}, n),$$

where $\text{Reg}(\mathcal{F}, n)$ is defined in terms of the surrogate loss ℓ'. Since $\mathbb{E}\left[\text{Reg}(\mathcal{F}, n)/n\right]$ goes to zero, we know that the average number of mistakes of the learner is arbitrarily close to the average surrogate loss of the best function from the reference class. This circumvents the (frustrating) impossibility result of learning 0-1 loss in the online setting beyond trivial classes. As mentioned, this is also what we do in the statistical learning setting for the purpose of computational efficiency.

Typical surrogate losses include the hinge loss $\ell(f, (x, y)) = \max\{1 - yf(x), 0\}$, the logistic loss $\ell(f, (x, y)) = \log(1 + e^{-yf(x)})$, and many more. Note that both losses are 1-Lipschitz (with respect to $f(x)$), and thus if $\mathcal{F}$ has a finite sequential fat-shattering dimension, the problem is indeed online learnable according to previous discussions, leading to a meaningful bound on the total 0-1 loss of the learner.

Summary. For regression problems with a G-Lipschitz loss, we have derived the following

$$\begin{aligned}
\mathcal{V}^{\text{seq}}(\mathcal{F}, n) &\le \sup_{\mathcal{P}\in\Delta(\mathcal{Z}^n)} \mathop{\mathbb{E}}_{z_{1:n}\sim\mathcal{P}} \left[\sup_{f\in\mathcal{F}} \frac{1}{n}\sum_{t=1}^{n} \left(\mathbb{E}_{z_t'\sim\mathcal{P}(\cdot|z_{1:t-1})}\left[\ell(f, z_t')\right] - \ell(f, z_t)\right)\right] \\
&\le 2\mathcal{R}^{\text{seq}}(\ell(\mathcal{F})) \le \mathcal{O}\left(G\ln^{3/2} n\right)\cdot \mathcal{R}^{\text{seq}}(\mathcal{F}) \\
&\le \mathcal{O}\left(G\ln^{3/2} n\right)\cdot \sup_{\boldsymbol{x}} \inf_{0\le\alpha\le 1}\left(4\alpha + \frac{12}{\sqrt{n}}\int_{\alpha}^{1}\sqrt{\ln \mathcal{N}_2(F|_{\boldsymbol{x}}, \delta)}d\delta\right) \\
&\le \mathcal{O}\left(G\ln^{3/2} n\right)\cdot \inf_{0\le\alpha\le 1}\left(4\alpha + \frac{12}{\sqrt{n}}\int_{\alpha}^{1}\sqrt{\text{sfat}(\mathcal{F}, \delta)\ln\left(\frac{2en}{\delta}\right)}d\delta\right).
\end{aligned}$$

It can be argued that this sequence of upper bounds is also tight and the sequential fat-shattering dimension is in some sense the right complexity measure. At this point, we have finished all discussions on statistical and online learnability for both classification and regression problems.

3 Online Algorithms for Finite Classes

We will now move on to the next main topic of this course: algorithm design for online learning. Recall that for statistical learning, all the problems we have discussed are learnable simply via ERM, if they are learnable at all. But for online learning, we have only characterized what determines learnability, and if the problem is learnable, we do not know how to learn yet. To think about how to design algorithms, we first recall the learning protocol:

For each $t = 1, \dots, n$,

1. learner (possibly randomly) selects $\widehat{y}_t \in \mathcal{D}$;
2. simultaneously the environment selects $z_t \in \mathcal{Z}$;
3. the learner suffers $\ell(\widehat{y}_t, z_t)$ and observes z_t.

Recall that $\mathcal{D}$ is the decision space of the learner, which is the same as the reference class $\mathcal{F}$ if the learner is proper, or a superset of $\mathcal{F}$ if the learner is improper (and thus more powerful). For an oblivious environment, z_t does not depend on the learner's decisions and can be equivalently seen as chosen ahead of the time before the learning protocol starts. On the other hand, for an adaptive environment, z_t could depend on $\widehat{y}_{1:t-1}$.

Since the first learnable class we showed is finite class, we start with the case when $\mathcal{F}$ is finite. Without loss of generality, assume that the value of the loss is always in $[0, 1]$. Then, based on previous discussions, there exists an algorithm with

$$\mathbb{E}[\text{Reg}(\mathcal{F}, n)] = \mathbb{E}\left[\sum_{t=1}^{n}\ell(\widehat{y}_t, z_t) - \inf_{f\in\mathcal{F}}\sum_{t=1}^{n}\ell(f, z_t)\right] \le \mathcal{O}\left(\sqrt{n\ln|\mathcal{F}|}\right). \tag{3}$$

Finding a concrete algorithm with this guarantee will be our goal for the rest of this lecture.

3.1 Warm-up

As the first step, we consider the special case of binary classification with 0-1 loss (so $\mathcal{Z} = \mathcal{X}\times\{-1, +1\}$ and $\mathcal{F}\subset\{-1, +1\}^{\mathcal{X}}$), and make a strong *realizable* assumption which posits that there is always a perfect classifier in $\mathcal{F}$, that is, $\inf_{f\in\mathcal{F}}\sum_{t=1}^{n}\mathbf{1}\{f(x_t)\neq y_t\} = 0$. This can be seen as an assumption on the expressiveness of $\mathcal{F}$ or equivalently as a constraint on the environment. What is a reasonable algorithm in this case?

An obvious solution is to pick $\widehat{y}_t$ to be any classifier in $\mathcal{F}$ that has made no mistakes yet so far. Then, whenever the learner makes a mistake, she has one less available choice. Since there is always a perfect classifier in $\mathcal{F}$, the learner makes at most $|\mathcal{F}| - 1$ mistakes, which is also her regret. This

regret bound is better than Equation (3) in terms of the dependence in n (in fact, it is independent of n), but exponentially worse in terms of $|\mathcal{F}|$.

Can we do better then? The issue of this simple algorithm is that in the worst case, every time we make a mistake, we can only eliminate one bad classifier. Ideally, we hope that a mistake will bring to us much more information, just like doing a binary search. It turns out that we can indeed do so by predicting the "majority vote" of the surviving classifiers, instead of just following any surviving one. This is the "Halving" algorithm, which at time t uses the following classifier:

$$\widehat{y}_t(x) = \text{sign}\left(\sum_{f \in \mathcal{F}'} f(x) \right), \quad \text{where } \mathcal{F}' = \left\{ f \in \mathcal{F} : \sum_{\tau=1}^{t-1} \mathbf{1}\left\{ f(x_\tau) \neq y_\tau \right\} = 0 \right\}.$$

Before analyzing the performance of this algorithm, note that this is an *improper* algorithm — $\widehat{y}_t$ is not from the class $\mathcal{F}$! Instead of writing down the improper function explicitly, it is often more convenient to equivalently change the learning protocol to: 1) environment first reveals x_t; 2) after seeing x_t, the learner makes a prediction -1 or $+1$ (in whatever way she likes); 3) environment reveals y_t. This setting is clearly more favorable to the learner if $\mathcal{D} = \mathcal{F}$, but it in fact does not give any extra power to an improper learner with $\mathcal{D} = \{-1, +1\}^{\mathcal{X}}$ since specifying how the learner predicts the label after seeing x_t is exactly the same as specifying an improper strategy from $\mathcal{D}$. For example, for the Halving algorithm, what the algorithm predicts at time t is simply the majority vote of the surviving classifiers on example x_t.

It is not hard too see that the Halving algorithm makes at most $\mathcal{O}(\log_2 |\mathcal{F}|)$ mistakes, since every time a mistake occurs, at least half of the classifiers are removed from the surviving set $\mathcal{F}'$. Consequently, the regret of the algorithm is also $\mathcal{O}(\log_2 |\mathcal{F}|)$, which is exponentially better than the naive strategy and is again n-independent. We remark that the realizable assumption is the key to getting a regret bound better than Equation (3).

3.2 General algorithm

So how can we go beyond the realizable setting, which is a rather strong assumption? First of all, we note that the two algorithms we discussed above are both *deterministic*. On the other hand, we now argue that without the realizable assumption, no deterministic algorithm can guarantee sublinear regret, even for a very simple problem where there is only one possible example x and $\mathcal{F}$ contains only two constant functions $f_+(x) = +1$ and $f_-(x) = -1$. Indeed, because the algorithm is deterministic, at the beginning of time t, its prediction on x is fixed and known to the environment already. Thus, if the environment always picks y_t to be the opposite of what the algorithm will predict, then the total loss of the learner is clearly just T, while one of f_- and f_+ must have total loss not larger than $T/2$, leading to linear regret that is at least $T/2$.

Also recall that in Lecture 4, we briefly mentioned that the analogue of ERM or the so-called follow-the-leader strategy $\widehat{y}_t = \text{argmin}_{f \in \mathcal{F}} \sum_{\tau=1}^{t-1} \ell(f, z_\tau)$ is not a good algorithm. The reason is clear now — this is a deterministic algorithm and will suffer linear regret even for simple problems.

Therefore, we need to shift our focus to randomized algorithms. Also, without the realizable assumption, it is clearly a bad idea to discard a classifier as soon as it makes a single mistake. Combining these two observations, one sees that a natural strategy is to randomly sample $f \in \mathcal{F}$ according to its previous performance — those with smaller cumulative loss so far should be sampled with higher probability. Indeed, this leads to a classic algorithm that works for general problems (not just classification with 0-1 loss):

$$\textbf{Hedge}\text{: at time } t\text{, sample } \widehat{y}_t \in \mathcal{F} \text{ with probability } \Pr\left[\widehat{y}_t = f\right] \propto \exp\left(-\eta \sum_{\tau=1}^{t-1} \ell(f, z_\tau) \right) \tag{4}$$

where $\eta > 0$ is a parameter of the algorithm (called learning rate or step size). This simple algorithm turns out to be fundamental and has broad applications in many areas (some of which will be discussed in future lectures). In fact, it was (re)discovered in different areas and has many different names, such as Hedge, multiplicative weight update, exponential weights, and so on.

Note that this is a proper algorithm. It can be viewed as doing a "soft" version of Follow-the-Leader because when η is infinity, the distribution simply puts all the mass on the current leader

$\operatorname{argmin}_{f\in\mathcal{F}} \sum_{\tau=1}^{t-1} \ell(f, z_\tau)$ and the algorithm becomes Follow-the-Leader. In fact, this mapping (from cumulative losses to an exponential distribution) is also known as the "softmax" function.

We will show that Hedge ensures exactly regret bound (3), giving an algorithmic certification that finite classes are online learnable. We use the following lemma that will be useful for future discussions as well.

Lemma 1. *For any $\ell_t \in \mathbb{R}^K$ and $\eta > 0$ such that $\eta\ell_t(i) \geq -1$ for all t and i, define $p_t \in \Delta(K)$ to be a distribution such that $p_t(i) \propto \exp\left(-\eta \sum_{\tau=1}^{t-1} \ell_\tau(i)\right)$. Then we have for any $i^\star \in [K]$,*

$$\sum_{t=1}^{n} \langle p_t, \ell_t\rangle - \sum_{t=1}^{n} \ell_t(i^\star) \leq \frac{\ln K}{\eta} + \eta \sum_{t=1}^{n}\sum_{i=1}^{K} p_t(i)\ell_t^2(i).$$

We defer the proof to the end of this section. With this lemma, it is straightforward to conclude the following.

Theorem 6. *For any environments (oblivious or adaptive), Hedge (defined in Equation (4)) with $\eta = \sqrt{\frac{\ln|\mathcal{F}|}{T}}$ ensures Equation (3).*

Proof. To apply Lemma 1, we set $K = |\mathcal{F}|$, rename the element of $\mathcal{F}$ by $1, \dots, K$, and set $\ell_t(i) = \ell(i, z_t)$, so that the learner exactly samples $\widehat{y}_t$ according to p_t as defined in Lemma 1. We then have

$$\mathbb{E}\left[\sum_{t=1}^{n} \ell(\widehat{y}_t, z_t)\right] = \mathbb{E}\left[\sum_{t=1}^{n} \langle p_t, \ell_t\rangle\right] \quad \text{and} \quad \mathbb{E}\left[\inf_{f\in\mathcal{F}} \sum_{t=1}^{n} \ell(f, z_t)\right] = \mathbb{E}\left[\min_{i^\star\in[K]} \sum_{t=1}^{n} \ell_t(i^\star)\right].$$

Applying Lemma 1 with expectation taken on both sides, using the fact $\ell_t(i) \leq 1$, and plugging in the particular learning rate η (which is optimal) proves Equation (3). □

Next we prove Lemma 1 based on a potential-based argument. While the proof (or maybe even the algorithm Hedge itself) might seem to come out of nowhere, in the future we will provide more explanation on this.

Proof of Lemma 1. Define $L_t = \sum_{\tau=1}^{t} \ell_\tau$ and "potential" $\Phi_t = \frac{1}{\eta} \ln\left(\sum_{i=1}^{K} \exp(-\eta L_t(i))\right)$. Now, we study how the potential evolves from time $t-1$ to time t:

$$\begin{aligned}
\Phi_t - \Phi_{t-1} &= \frac{1}{\eta} \ln\left(\frac{\sum_{i=1}^{K} \exp(-\eta L_t(i))}{\sum_{i=1}^{K} \exp(-\eta L_{t-1}(i))}\right) = \frac{1}{\eta} \ln\left(\sum_{i=1}^{K} p_t(i) \exp(-\eta\ell_t(i))\right) \\
&\leq \frac{1}{\eta} \ln\left(\sum_{i=1}^{K} p_t(i) \left(1 - \eta\ell_t(i) + \eta^2\ell_t^2(i)\right)\right) \qquad (e^{-y} \leq 1 - y + y^2 \text{ for all } y \geq -1) \\
&= \frac{1}{\eta} \ln\left(1 - \eta \langle p_t, \ell_t\rangle + \eta^2 \sum_{i=1}^{K} p_t(i)\ell_t^2(i)\right) \\
&\leq -\langle p_t, \ell_t\rangle + \eta \sum_{i=1}^{K} p_t(i)\ell_t^2(i). \qquad (\ln(1+y) \leq y)
\end{aligned}$$

Summing over t, telescoping, and rearranging gives

$$\begin{aligned}
\sum_{t=1}^{n} \langle p_t, \ell_t\rangle &\leq \Phi_0 - \Phi_n + \eta \sum_{t=1}^{n}\sum_{i=1}^{K} p_t(i)\ell_t^2(i) \\
&\leq \Phi_0 - \frac{1}{\eta} \ln\left(\exp(-\eta L_n(i^\star))\right) + \eta \sum_{t=1}^{n}\sum_{i=1}^{K} p_t(i)\ell_t^2(i) \\
&= \frac{\ln K}{\eta} + L_n(i^\star) + \eta \sum_{t=1}^{n}\sum_{i=1}^{K} p_t(i)\ell_t^2(i).
\end{aligned}$$

Further rearranging finishes the proof. □