Learning Theory

John Augustine (IIT Madras)

Krishna Palem (Rice University)

Source:
These slides are based on source material (including notation, concepts, and presentation approach) from "An introduction to Computational Learning theory" by Michael J. Kearns and Umesh V. Vazirani and "Analysis of Boolean Functions" by Ryan O'Donnell. The PAC learning game is borrowed from Jeremy Kun's blog www.jeremykun.com/primers

Acknowledgement: We thank Shreyas Shetty M (IIT Madras) for collaboration that resulted in these slides.

Topics in this Lecture

1. Recap
2. PAC learning of Boolean Functions
   - 2.a Definitions and Notations
   - 2.b Theorem
   - 2.c Proof
3. A PAC learnable task
   - 3.a PAC model setup
   - 3.b Proof of PAC learnability

Recap and Some Useful Background

Recap

- Probably approximately Correct(PAC) learning as a model.
- The PAC model captures conditions for *efficient* learning.
- Given a learning task and the model, we can derive formal guarantees about *learnability*.
  - Preferably the absolute minimum number of examples needed to learn well.
    - Cost.
  - The preferably very low probability of our answer being correct within a desired error bound.
    - Quality.
- We discussed a central theorem which characterized an algorithm for learning "sparse" Boolean functions.

In this lecture

- We will first give a formal proof of this theorem.
- We will then follow up with PAC learning of general functions.
  - Give an example of a task that is efficiently PAC learnable.

We start with the definition of an inner product of two functions

- Formally, we define the *inner product* of a pair of boolean functions $f, g: \{-1, +1\}^n \to \mathbb{R}$ as follows

$$\langle f, g \rangle = \frac{1}{2^n} \sum_{x \in \{-1,+1\}^n} f(x) g(x) .$$

- This is equivalent to saying that the inner product is the expected value of the product of f and g when the input string is drawn uniformly at random.

$$\langle f, g \rangle = E_{x \sim \{-1,+1\}^n} [f(x) g(x)] \; (*)$$

  - As always $x \sim \{-1, +1\}^n$ means x is chosen uniformly at random from $\{-1, +1\}^n$

A couple of additional definitions and known facts that we will use.

$$f: \{-1, +1\}^n \to \{-1, +1\}$$

- Parseval's theorem

$$\sum_{S \subseteq [1 \cdots n]} \hat{f}(S)^2 = 1$$

- Parseval's theorem tells us that Fourier coefficients cannot assume arbitrary values.
  - There is structure since the squared sum adds up to 1 for Boolean functions with range $\{\pm 1\}$.
  - Recall from (*) $\langle f, g \rangle = E_{x \sim \{-1,+1\}^n}[f(x)g(x)]$.
  - Replacing g with f above we have $\langle f, f \rangle = E_{x \sim \{-1,+1\}^n}[f(x)f(x)] = E_{x \sim \{-1,+1\}^n}[f(x)^2]$.
  - $E_{x \sim \{-1,+1\}^n}[f(x)^2] = E_{x \sim \{-1,+1\}^n}[1] = 1$.
  - Also, $\langle f, f \rangle = \sum_{S \subseteq [1 \cdots n]} \hat{f}(S)^2$
    - Please refer to Page nos. 8 and 9, following Proposition 1.8 (Chapter 1), "Analysis of Boolean Functions" by Ryan O'Donnell.
    - For access to the book please go to http://get.analysisofbooleanfunctions.org/

Restating ϵ-concentration through *spectral sampling*

- The *fourier weight* of $\boldsymbol{f}$ on a set S is $\hat{\boldsymbol{f}}(\boldsymbol{S})^{\boldsymbol{2}}$.
- *Spectral sample* $\boldsymbol{S_f}$ is the probability distribution of all possible subsets $S \subseteq [1 \cdots n]$.`
- S has probability $p_S = \hat{\boldsymbol{f}}(\boldsymbol{S})^{\boldsymbol{2}}$.
- Valid probability since $\hat{\boldsymbol{f}}(\boldsymbol{S})^{\boldsymbol{2}}$ is positive and $\sum_{S \subseteq [1 \cdots n]} \hat{\boldsymbol{f}}(\boldsymbol{S})^{\boldsymbol{2}} = 1$.
- As before, let $\boldsymbol{\mathcal{F}}$ be a collection of subsets $S \subseteq [n]$.
- We can now reinterpret the Fourier spectrum of $f: \{-1,1\}^n \to \mathbb{R}$ ϵ-concentrated on $\boldsymbol{\mathcal{F}}$ as follows
  - $\sum_{\boldsymbol{S \subseteq [n]} \\ \boldsymbol{S \notin \mathcal{F}}} p_S \leq \boldsymbol{\epsilon}$.
- ϵ concentration gives the probability of coefficients associated with subsets S from $\boldsymbol{\mathcal{F}}$ through those which are *not* in $\boldsymbol{\mathcal{F}}$.

Proving the theorem about PAC learning of Boolean functions

Mapping definitions from the PAC model to the Boolean domain.

- Recall that we had defined an error function $error(h)$ to measure the error between the hypothesis h and the true concept c.
- We had defined error as the $\mathbb{E}[(h(x) - g(x))^2]$
- We will now define the distance(another form of error) between hypothesis h and f as
  - $\boldsymbol{dist(h, f) = Pr_{x \sim \{-1,1\}^n}[h(x) \neq f(x)]}$
  - Note that x is chosen uniformly at random from $\{-1,1\}^n$.
  - It can be shown that these two definitions are mathematically related, and we will use this relationship henceforth.

Intuition about the relationship between the two notions of error

- Let $\mathbb{I}$ be an *indicator function* which is 1 when $f(x) \neq h(x)$ and 0 otherwise.
- By definition, $dist(f,h) = Pr_{x\sim\{-1,1\}^n}[h(x) \neq f(x)] = \mathbb{E}_{[x\sim\{-1,1\}^n]}[\mathbb{I}_{[f(x)\neq h(x)]}]$
- Since $|f(x) - h(x)|^2 \geq 1$ whenever $f(x) \neq h(x)$ or equivalently when $\mathbb{I}_{[f(x)\neq h(x)]} = 1$ (and both values are equal to zero otherwise)
- $dist(f,h) = \mathbb{E}_{[x\sim\{-1,1\}^n]}[\mathbb{I}_{[f(x)\neq h(x)]}] \leq \mathbb{E}_{[x\sim\{-1,1\}^n]}[\,|f(x) - h(x)|^2]$
  - New definition of error is tighter or ≤ the previous definition.

An interesting result

Theorem. Assume learning algorithm A has access to randomly chose examples as inputs to $f: \{-1,1\}^n \to \{-1,1\}$.
Suppose that A can (through an oracle) identify a collection $\boldsymbol{\mathcal{F}}$ of subsets on which f's Fourier spectrum is $\frac{\epsilon}{2}$ - concentrated. Then using poly($|\mathcal{F}|$, n, $1/\epsilon$) additional time, algorithm A can with high probability output a hypothesis h that is ϵ-close to f.

We will use $dist(f,h) = \Pr_{\{x \sim \{-1,1\}^n\}} f(x) \neq h(x)$

Proof structure

- Step 1: Develop an algorithm for approximating the Fourier coefficient $\tilde{f}$ on a given set S.
- Step 2: Devise a method to build a provably good approximation for the true function $f(\boldsymbol{x})$ building on Step 1.
- Step 3: Build on step 2 and prove the theorem.
- Note: *To aid Steps 1 and 2, we will quote without proof a method to determine the collection of sets* $\boldsymbol{\mathcal{F}}$.

Step 1

Step 1: Develop an algorithm for approximating the Fourier coefficient on a given set S.

Proposition 1. Given access to randomly chosen examples $f:\{-1,1\}^n \to \{-1,1\}$ in the form $(\boldsymbol{x}\ f(\boldsymbol{x}))$, a randomized algorithm can be constructed which takes an input $S \subseteq [1 \dots n]$, $0 < \delta, \epsilon \leq \frac{1}{2}$, and outputs an estimate $\tilde{f}(S)$ for $\hat{\boldsymbol{f}}(S)$ that satisfies

$$\left|\tilde{f}(S) - \hat{f}(S)\right| \leq \epsilon$$

with probability at least δ. Furthermore, its running time is poly$(n, 1/\epsilon)$.log$(1/\delta)$.

Proof

- By definition we have $\hat{f}(S) = E_x[f(x)x^S]$.
  - We are given examples $(x, f(x))$ where x is chosen uniformly at random.
  - Using these, we can compute $f(x)x^S \in \{-1, 1\}$ and get an empirical estimate of $E_x[f(x)x^S]$.
- How many examples would be needed to get a good estimate
  - This means error bounded by $\pm\epsilon$ is true with high probability $\geq 1 - \delta$?
- An application of *Chernoff bounds* that you have seen gives us the desired number of examples *M*.

Recalling Chernoff bounds

- Chernoff Bounds give us a way to bound sums of i.i.d. (independent and identically distributed) random variables.
- Chernoff Bound:
  - Let $X_1, X_2, .., X_n$ be t independent random variables taking values in the interval $[0,1]$, let $X = (\sum_i X_i)/t$, and $\mu = \mathbb{E}[X]$, then

$$\Pr[|X - \mu| \geq \epsilon] \leq 2\exp(-\frac{t\epsilon^2}{4})$$

Proof

- We can show that we need $\mathrm{M} = O(\frac{1}{\epsilon^2}\log\frac{1}{\delta})$ examples to get the error bounded by $\pm\epsilon$ with high probability $1-\delta$
- You will be given a detailed exercise to complete this computation.
- Hint: Note that $f(x)x^S \in \{-1,1\}$ (why?) and let

$$Y = f(x)x^S \qquad \mathbb{E}[Y] = \hat{f}(S)$$

- Let $Y' = \frac{1-Y}{2}$, observe that $Y' \in [0,1]$ and apply the Chernoff Bound form given in the previous slide to obtain the number of examples needed.

The claim about running time

- Now, assume that one can access a sample from our oracle in $O(1)$ time.
- This will result in a running time of $poly\left(n, \frac{1}{\epsilon}\right).\log\left(\frac{1}{\delta}\right)$
  - Since we assumed $O(1)$ time for each example we need $O(\frac{1}{\epsilon^2}\log\frac{1}{\delta})$ time for obtaining samples.
  - For each sample $(x, f(x))$, $f(x)x^s$ can take $O(n)$ time. This is primarily due to x^s computation.
  - Resulting in a total of $O(\frac{n}{\epsilon^2}\log\frac{1}{\delta})$ which can be written as $poly\left(n, \frac{1}{\epsilon}\right).\log\left(\frac{1}{\delta}\right)$

Step 2

We need to use the concept of an inner Product of Two Functions

- Formally, we define the *inner product* of a pair of boolean functions $f, g: \{-1, +1\}^n \to \mathbb{R}$ as follows

$$\langle f, g \rangle = \frac{1}{2^n} \sum_{x \in \{-1,+1\}^n} f(x) g(x) .$$

- This is equivalent to saying that the inner product is the expected value of the product of f and g when the input string is drawn uniformly at random.

$$\langle f, g \rangle = E_{x \sim \{-1,+1\}^n}[f(x) g(x)]$$

  - As always $x \sim \{-1, +1\}^n$ means x is chosen uniformly at random from $\{-1, +1\}^n$

Two more definitions

- Notation: $f - g$ is $f(x) - g(x)$
- Using the definition of an inner product, we define the ℓ_2 norm as

$$\|f\|_2^2 = < f, f >$$

- Consider two Boolean functions f, g.
- We can define the difference or distance between them as $\|f - g\|_2^2$
- Using the definition of the ℓ_2 norm, we have

$$\|f - g\|_2^2 = < f - g, f - g >$$

  - From (**), we know that the inner product can be written as $\langle f, f \rangle = \sum_{S \subseteq \{1,\dots,n\}} \hat{f}(S)^2$ namely using fourier coefficients instead of function values $f(x)$.
  - This alternate form will be used by us in the sequel.

- $sgn(f(x))$ is the sign function.
- $sgn(f(x)) = \begin{cases} 1 & f(x) \geq 0 \\ -1 & f(x) < 0 \end{cases}$

Step 2: Devise a method to build a provably good approximation for the true function $f(x)$ building on Step 1. Shreyas: please check phrasing of propostion carefully.

Proposition 2. Suppose that $f: \{-1,1\}^n \to \{-1,1\}$ and $g: \{-1,1\}^n \to \mathbb{R}$ satisfy $\|f - g\|_2^2 \leq \epsilon$.
Let $h: \{-1,1\}^n \to \{-1,1\}$ be defined by $h(x) = sgn(g(x))$, with $sgn(0)$ chosen arbitrarily from $\{-1,1\}$. Then $dist(f,h) \leq \epsilon$.

Note: g is an approximation to f

What proposition 2 means informally

- g is an intermediate function that we construct.
  - Notice that the range of g is $\mathbb{R}$ and not $\{-1.+1\}$.
- We wish to build a hypothesis h that is a Boolean function with range $\{-1, +1\}$.
- We apply an appropriate transformation (sgn) to our "crude" intermediate function g to derive h.

Proof

- Whenever $f(x) \neq sgn(g(x))$, $|f(x) - g(x)|^2 \geq 1$.
- Therefore we have,

$dist(f,h) = \Pr_x[f(x) \neq h(x)]$ *(by definition)*

$= \mathbb{E}_{[x\sim(-1,1)^n]}[\mathbb{I}_{[f(x)\neq sgn(g(x))]}]$

(Known fact: Expectation of indicator R.V. is equal to probability of $\{f(x) \neq sgn(g(x))\}$ *and* $h(x) = sgn(g(x))$ *).*

$\leq \mathbb{E}_{[x\sim(-1,1)^n]}[\,|f(x) - g(x)|^2]$

(since, $|f(x) - g(x)|^2 \geq 1$ *whenever* $\mathbb{I}$ *is 1)*

$= \|f - g\|_2^2$ (ℓ_2 norm definition)

$\leq \epsilon$ (from statement of the proposition)

Therefore, $dist(f,h) \leq \epsilon$

- This proves proposition 2.

Step 3

Completing the proof

An interesting result

Theorem. Assume learning algorithm A has access to randomly chose examples as inputs to $f:\{-1,1\}^n \to \{-1,1\}$.
Suppose that A can (through an oracle) identify a collection $\mathcal{F}$ of subsets on which f's Fourier spectrum is $\frac{\epsilon}{2}$ - concentrated. Then using poly($|\mathcal{F}|$, n, $1/\epsilon$) additional time, algorithm A can with high probability output a hypothesis h that is ϵ-close to f.

We will use $dist(f,h) = \Pr_{[x\sim\{-1,1\}^n]} f(x) \neq h(x)$

Step 3

- For each $S \in \mathcal{F}$, algorithm A can, using Proposition 1, get an estimate $\tilde{f}(S)$ for $\hat{f}(S)$ which satisfies
  - $|\hat{f}(S) - \tilde{f}(S)| \leq (\sqrt{\epsilon}/2\sqrt{|\mathcal{F}|})$
    - Important note: The entire RHS of this inequality corresponds to ϵ from Proposition 1 above.
    - Thus ϵ as used in the main theorem and in the proof in step 3 (now) are identical, and different from the symbol ϵ used in Proposition 2. to represent a different variable.
  - With probability at least $(1 - \frac{1}{10|\mathcal{F}|})$.
    - Choose the δ from Proposition 1 as above.
  - These numbers are chosen so as to simplify the computations below.

Step 3

- **Known fact:** Union bound: Probability of union of a finite set of events is less than or equal to the sum of probabilities of the individual events.

$$P(A_1 \cup A_2 \cup ..\cup A_n) \leq P(A_1) + P(A_2) + \ ... + P(A_n)$$

- Our design states that for each $S \in \mathcal{F}$, we have the probability of the estimate satisfying being correct that is $|\hat{f}(S) - \tilde{f}(S)| \leq (\sqrt{\epsilon}/2\sqrt{|\mathcal{F}|})$ is at least $(1 - \frac{1}{10|\mathcal{F}|})$.
- On application of the union bound we have, with probability at least 9/10 that all $|\mathcal{F}|$ estimates have the desired accuracy.

Step 3 (contd.)

- Now, consider the following constructive step:
  - Our learning algorithm, A forms $g = \sum_{S \in \mathcal{F}} \hat{f}(S) x^S$ and outputs $h = sgn(g)$.
- Given g we have,

Fact: $\|f - g\|_2^2 \leq \sum_{S \in \mathcal{F}} \left(\hat{f}(S) - \hat{f}(S)\right)^2 + \sum_{S \notin \mathcal{F}} \hat{f}(S)^2$

(using the form of ℓ_2 norm from (**))

$\leq \sum_{S \in \mathcal{F}} \left(\frac{\sqrt{\epsilon}}{2\sqrt{|\mathcal{F}|}}\right)^2 + \frac{\epsilon}{2} \leq \frac{\epsilon}{4} + \frac{\epsilon}{2} < \epsilon$ with probability 9/10.

(The first term in the sum follows from the fact that $|\hat{f}(S) - \hat{f}(S)| \leq (\sqrt{\epsilon}/2\sqrt{|\mathcal{F}|})$ by design with the probabilistic assertion following from the union bound.)

(The second term $\frac{\epsilon}{2}$ is immediate from the definition of ϵ-concentration and the hypothesis of our theorem.)

- From the fact above and proposition 2 we have, $\boldsymbol{dist(f, h) \leq \epsilon}$ as desired.
- This completes the proof.

Finding a suitable $\mathcal{F}$

- Our previous argument, assumes that we know which are the sets that belong to the collection $\mathcal{F}$.
- All that remains is an efficient procedure to determine the collection $\mathcal{F}$.
- A well known Goldreich-Levin algorithm gives us a polynomial time algorithm to find $\mathcal{F}$ on which the function f is ϵ – concentrated.
  - Please refer to Section 3.5 of "Analysis of Boolean functions", by Ryan O'Donnell

An interesting result

Theorem. Assume learning algorithm A has access to randomly chose examples as inputs to $f:\{-1,1\}^n \to \{-1,1\}$.
Suppose that A can (through an oracle) identify a collection $\mathcal{F}$ of subsets on which f's Fourier spectrum is $\frac{\epsilon}{2}$ - concentrated. Then using poly($|\mathcal{F}|$, n, $1/\epsilon$) additional time, algorithm A can with high probability output a hypothesis h that is ϵ-close to f.

We will use $dist(f,h) = \Pr_{[x\sim\{-1,1\}^n]} f(x) \neq h(x)$

Running time

- This requires poly($|\mathcal{F}|, n, 1/\epsilon$) time.
  - From proposition 1, each step to determine a $\tilde{f}(S)$ takes $poly\left(n, \frac{\sqrt{|\mathcal{F}|}}{\epsilon}\right) . \log(|\mathcal{F}|)$ time.
    - Simple substitution and will be an exercise.
  - We have $|\mathcal{F}|$ such operations and hence we get poly($|\mathcal{F}|, n, 1/\epsilon$)

PAC learnable game

Let's play a game to firmly understand PAC learning in general.

- Consider a game between Alice and Bob.
- Alice generates a number $\boldsymbol{x}$ at random in some fixed way (has a fixed distribution for picking $\boldsymbol{x}$).
- Alice has a fixed interval $[\boldsymbol{a}, \boldsymbol{b}]$ in her mind.
- Once $\boldsymbol{x}$ is generated, Alice gives out a pair.
- The pair consists of $\boldsymbol{x}$ and 1 if $\boldsymbol{x}$ lies in the interval and 0 otherwise.
- **Goal:** Bob sees a sequence of such pairs and tries to determine $\boldsymbol{a}$ and $\boldsymbol{b}$.

Some observations

- Bob cannot determine the exact interval if $\boldsymbol{a}, \boldsymbol{b}$ are real, since he only gets a finite number of examples.
- Whatever interval Bob determines, at the end, we can test that interval against the number generating scheme of Alice i.e. $\boldsymbol{D}$.
  - $\boldsymbol{D}$ is the target distribution from which the examples are drawn
- In other words, we can compute the probability that Bob's interval will give an incorrect label if Alice were to keep generating the numbers indefinitely.
- **Success criterion:** If this probability is small, then Bob has "learned" the interval.

PAC learnability

- Consider the following natural (trivial) algorithm for learning:
  - Bob asks Alice for a *sufficient* number of sample points, say m.
  - He then takes the smallest and the biggest positive example and use those points as the endpoints of the hypothesis interval.
- Intuitively, the more points Bob sees, the closer to predicting the actual interval he is.
- **Question:** What is the optimal value of m for Bob to have "learned" the interval ?
  - We expect Bob to have learnt the interval with a high probability and hence m is the (smallest is most desirable) size parameter yielding this result.

Proof of PAC learnability

- Let the set of all indicator functions of intervals be our concept class.
  - In other words, consider intervals $[x, y]$, and their corresponding indicator functions, $\mathbb{I}$ which equals 1 for inputs within the interval and 0 otherwise.
- Let us fix any distribution D over the real line and consider m samples.
- Further, let Bob pick the maximum and minimum of the *positive* examples
  - in that they have their associated indicator variables have a value of 1 as determined by Alice's output
  - and let the interval derived be $I = [a_1, b_1]$.

Proof of PAC learnability

- Let the target concept, i.e. the interval in Alice's mind be $\boldsymbol{J} = [\boldsymbol{a}, \boldsymbol{b}]$.
- Observe that $\boldsymbol{I} \subseteq \boldsymbol{J}$. This is because, $\boldsymbol{b_1} \leq \boldsymbol{b}$ and $\mathbf{a} \leq \boldsymbol{a_1}$
  - Since the maximum positive example is less than (or equal to) the actual upper limit.
  - And the minimum positive example is greater than (or equal to) the actual lower limit.

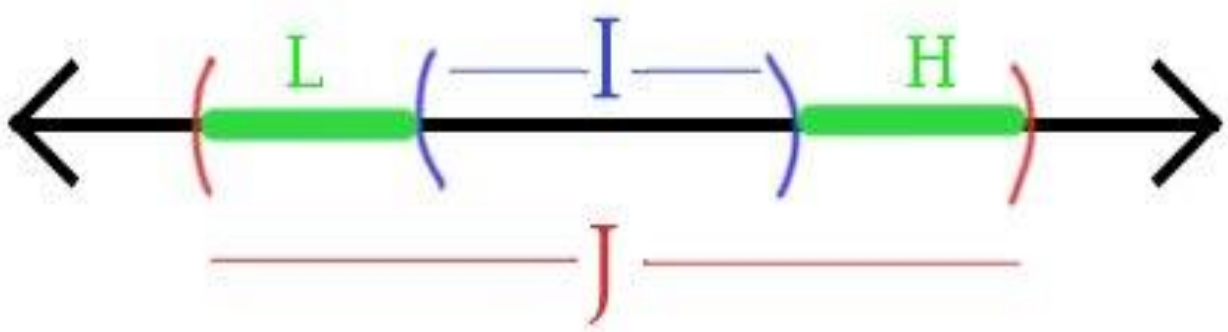
Proof of PAC learnability

- Probability that our hypothesis produces an incorrect label is equal to the probability that $\boldsymbol{D}$ produces a positive example in the two intervals.
  - $\boldsymbol{L = [a, a_1]}$ $\boldsymbol{H = [b_1, b]}$ *
- Error is at most the sum of probabilities of positive examples in L and H i.e.
  - $\boldsymbol{error(h) \leq P_{x \sim D}(x \in L) + P_{x \sim D}(x \in H)}$
    - $\boldsymbol{x}$ is drawn at random from $\boldsymbol{D}$

* L and H will be referred to as lower and upper intervals.

Proof of PAC learnability

Pictorially the intervals now look like

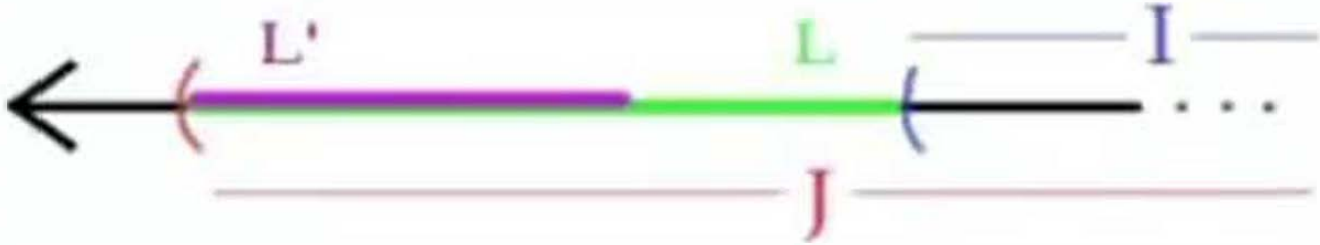


Proof of PAC learnability

- If the probability associated with each of the intervals is smaller than $\frac{\epsilon}{2}$, then we are done since their sum is less than ϵ.
- Let us focus on the interval L.
  - By symmetry, our arguments will also hold for H.

Proof of PAC learnability

- Define L' as follows
  - L' is the interval $[a, y]$, which is large enough such that the probability of a positive example drawn from L' under D is exactly equal to $\frac{\epsilon}{2}$.
  - If $L \subset L'$, we are done since the first term in the error probability $error(h) \leq \frac{\epsilon}{2}$.



Proof of PAC learnability

- We will now consider $\boldsymbol{L' \subset L}$.
- Probability of a sample not being in $\boldsymbol{L'}$ is $\boldsymbol{1 - \frac{\epsilon}{2}}$.
- We have $\boldsymbol{m}$ such independent samples.
- Therefore, the probability of missing $\boldsymbol{L'}$, i.e. chosen L contributes error greater than $\frac{\epsilon}{2}$ is

$$\boldsymbol{P_D(L' \subset L) \leq \left(1 - \frac{\epsilon}{2}\right)^m}$$

- Similar argument holds for $\boldsymbol{H}$ and hence we have
  - $\boldsymbol{error(h) = P(\, error(I) > \epsilon) \leq 2\left(1 - \frac{\epsilon}{2}\right)^m}$

Proof of PAC learnability

- We want the probability to be smaller than $\boldsymbol{\delta}$.
- Therefore, $\mathbf{2}\left(\mathbf{1}-\frac{\boldsymbol{\epsilon}}{\mathbf{2}}\right)^{\boldsymbol{m}} \leq \boldsymbol{\delta}$
- Using the known fact that $(1-x) \leq e^{-x}$, we get
  - $\mathbf{2}\boldsymbol{e}^{-\frac{\boldsymbol{\epsilon m}}{\mathbf{2}}} \leq \boldsymbol{\delta}$
  - This leads us to $\boldsymbol{m} \geq (\frac{\mathbf{2}}{\boldsymbol{\epsilon}}\mathbf{log}\left(\frac{\mathbf{2}}{\boldsymbol{\delta}}\right))$
- We see that for a given $\boldsymbol{\epsilon}, \boldsymbol{\delta}$ we can choose $\boldsymbol{m}$ to get the desired error bounds.
- Hence the game is PAC learnable.

Thank You!