Advanced Machine Learning

Bogdan Alexe,

bogdan.alexe@fmi.unibuc.ro

University of Bucharest, 2nd semester, 2021-2022

Uniform convergence

If ε-representative samples allows us to learn as good as possible, we can agnostically PAC learn if we can guarantee that we will almost always get (with probability 1 – δ) ε-representative sample.

Definition (*uniform convergence*)

A hypothesis class $\mathcal{H}$ has the *uniform convergence property* wrt a domain $\mathcal{Z}$, loss function ℓ if:

- there exists a function $m_H^{UC} : (0,1)^2 \rightarrow \mathbf{N}$
- such that for all $(\varepsilon, \delta) \in (0,1)^2$
- and for any probability distribution $\mathcal{D}$ over $\mathcal{Z}$

if S is a sample of $m \geq m_H^{UC}(\varepsilon, \delta)$ examples drawn i.i.d. according to $\mathcal{D}$, then, with probability of at least 1 – δ, S is ε-representative.

The term *uniform* refers to having a fixed sample size that works for all members of $\mathcal{H}$ and over all possible probability distributions $\mathcal{D}$ over the domain $\mathcal{Z}$

Finite classes are agnostic PAC learnable

Theorem

Let $\mathcal{H}$ be a finite hypothesis class, let $\mathcal{Z}$ be a domain and let $\ell: \mathcal{H} \times \mathcal{Z} \to [0,1]$ be a loss function. Then $\mathcal{H}$ has the uniform convergence property with sample complexity:

$$m_{\mathcal{H}}^{UC}(\epsilon, \delta) \leq \left\lceil \frac{\log(2|\mathcal{H}|/\delta)}{2\epsilon^2} \right\rceil$$

Moreover, the class $\mathcal{H}$ is agnostically PAC learnable using the ERM paradigm with sample complexity:

$$m_{\mathcal{H}}(\epsilon, \delta) \leq m_{\mathcal{H}}^{UC}(\epsilon/2, \delta) \leq \left\lceil \frac{2\log(2|\mathcal{H}|/\delta)}{\epsilon^2} \right\rceil$$

Beyond the result

By going from realizability to agnostic, we go:

- from $m_{\mathcal{H}}(\epsilon, \delta) \leq \left\lceil \frac{\log(|\mathcal{H}|/\delta)}{\epsilon} \right\rceil$
- to $m_{\mathcal{H}}(\epsilon, \delta) \leq m_{\mathcal{H}}^{UC}(\epsilon/2, \delta) \leq \left\lceil \frac{2\log(2|\mathcal{H}|/\delta)}{\epsilon^2} \right\rceil$

The denominator goes from ε to ε^2, which means that for the same of accuracy the minimal sample size grows by a factor of $1/\varepsilon$.

The No-Free-Lunch theorem

Prior knowledge

Empirical Risk Minimization (ERM) = learning paradigm that returns a predictor h that minimizes $L_S(h)$, S –training sequence of examples sampled i.i.d. from an unknown distribution $\mathcal{D}$ over a domain $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$

- ERM might overfit if we are not careful

To guard against overfitting we introduced some prior knowledge (inductive bias)

- hypothesis class = $\mathcal{H} \subset \mathcal{Y}^{\mathcal{X}}$
- revised ERM rule: apply the ERM learning paradigm over $\mathcal{H}$
- for the training sample S, the $\text{ERM}_{\mathcal{H}}$ learner chooses a predictor $h \in \mathcal{H}$ with the lowest possible error over S:

$$\text{ERM}_{\mathcal{H}}(S) \in \operatorname*{argmin}_{h \in \mathcal{H}} L_S(h),$$

Universal learner?

Do we need prior knowledge ($\mathcal{H} \subset \mathcal{Y}^{\mathcal{X}}$) for the success of learning? Consider $\mathcal{H}$ the set of all functions from $\mathcal{X}$ to $\mathcal{Y}$, $\mathcal{H} = \{h: \mathcal{X} \to \mathcal{Y}\}$. This class represents lack of prior knowledge: every member of it is a good candidate.

Maybe there exists some kind of universal learner = a learner who has no prior knowledge about a certain task and is ready to be challenged by any task. A specific task is defined by an unknown distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$, where the goal of the learner is to find a predictor $h: \mathcal{X} \to \mathcal{Y}$ whose risk $L_{\mathcal{D}}(h)$ is small enough.

Does there exists a learning algorithm A and a training set size *m* such that for every distribution $\mathcal{D}$, if A receives *m* i.i.d. samples from $\mathcal{D}$ it will output with high confidence a predictor *h* that has a small error?

The No-Free-Lunch theorem states that no such universal learner exists. For binary classification prediction tasks ($\mathcal{Y} = \{0,1\}$) for every learner there exists a distribution on which it fails (the learner will output a hypothesis with large generalization error)

The No-Free-Lunch theorem

Theorem (No-Free-Lunch)

Let A be any learning algorithm for the task of binary classification with respect to the 0−1 loss over a domain $\mathcal{X}$. Let m be any number smaller than $|\mathcal{X}|/2$, representing a training set size.

Then, there exists a distribution $\mathcal{D}$ over $\mathcal{X} \times \{0,1\}$ such that:

1. there exists a function $f : \mathcal{X} \to \{0, 1\}$ with $L_{\mathcal{D}}(f) = 0$.
2. with probability of at least 1/7 over the choice of $S \sim \mathcal{D}^m$ we have that $L_{\mathcal{D}}(A(S)) \geq 1/8$.

- *In other words, for every learning algorithm A there are cases for which this algorithm will fail whereas there is another learner (e.g. a trivial successful learner in this case would be an ERM learner with the hypothesis class $\mathcal{H} = \{f\}$, or more generally, ERM with respect to any finite hypothesis class that contains f and whose size satisfies the equation $m \geq 8\log(7|H|/6)$ that solves the task. It simply means that an adversary can use the fact that A has no clue what happens on the other half of the domain. We cannot learn perfectly without the proper background knowledge.*

Intuition of the proof

Let $C \subseteq \mathcal{X}$, such that $|C| = 2m$, $C = \{c_1, c_2, \ldots, c_{2m}\}$.

Any learning algorithm that observes only *m* training examples = half of the instances in $C = \{c_1, c_2, \ldots, c_m\}$ has no information on what should be the labels of the rest of the instances in $C = \{c_{m+1}, c_{m+2}, \ldots, c_{2m}\}$.

We expect that on the other half it will predict correctly half of them. Over all it will do on average 25% mistakes (50% mistakes on the unseen 50% data). There exists a distribution $\mathcal{D}_i$ over $C \times \{0, 1\}$ on which the learning algorithm will make $\geq 25\%$ mistakes.

Apply Markov inequality to show the final result:

$$\underset{S \sim D^m}{P}\left(L_D(A(S)) \geq \frac{1}{8}\right) \geq \frac{1}{7}$$

Idea of the proof

To prove the theorem, we will construct a set of distributions such that there exist a distribution on which A will fail. The basic ingredients are:

- let C $\subseteq$ X, such that |C| = 2m, C = $\{c_1, c_2, \ldots, c_{2m}\}$
- there exist T = 2^{2m} functions from C to {0, 1}
- denote these functions by $f_1, \ldots, f_T$.
  - f_1 assigns to all elements of C the value 0: $f_1(c_i) = 0$, i = 1, 2, …, 2m
  - f_2 assigns to all elements of C value 0 except c_1 : $f_2(c_1) = 1, f_2(c_i) = 0$, i = 2, 3, … , 2m
  - f_3 assigns to all elements of C value 0 except c_2 : $f_2(c_2) = 1, f_2(c_i) = 0$, i = 1, 3, … , 2m
  - ….
  - f_T assigns to all elements of C value 1 : : $f_T(c_i) = 1$, i = 1, 2, …, 2m
- for each function f_i define the distribution $\mathcal{D}_i$ over C × {0, 1} such that:

$$\mathcal{D}_i(\{(x,y)\}) = \begin{cases} 1/|C| & \text{if } y = f_i(x) \\ 0 & \text{otherwise.} \end{cases}$$

- $\mathcal{D}_i$ is perfect for f_i. It will only generate samples (x, y) on which f_i is correct (y = f_i(x)). Hence $L_{D_i}(f_i) = 0$.

Idea of the proof

For every learning algorithm A that receives a sample S of m elements of $C \times \{0, 1\}$ and returns a function $A(S) : C \to \{0, 1\}$ we have:

$$\max_{i \in [T]} \mathop{\mathbb{E}}_{S \sim \mathcal{D}_i^m} [L_{\mathcal{D}_i}(A(S))] \geq 1/4.$$

This part is the main core of the proof. It is very technical, we skip it, see the details in the book. The intuition tells us that on average we make errors on 25% of examples. But now we are interest in the worse case scenario (the maximum error).

Thus, we have (by choosing a maximizing i for the above inequality) that for every learning algorithm A' that receives m examples from $\mathcal{X} \times \{0, 1\}$ there exist a function $f : \mathcal{X} \to \{0, 1\}$ and a distribution $\mathcal{D}$ over $\mathcal{X} \times \{0, 1\}$, such that $L_{\mathcal{D}}(f) = 0$ and:

$$\mathop{\mathbb{E}}_{S \sim \mathcal{D}^m} [L_{\mathcal{D}}(A'(S))] \geq 1/4.$$

Idea of the proof

$$\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}[L_{\mathcal{D}}(A'(S))] \geq 1/4.$$

Use the Markov's inequality to derive the following lemma:

Lemma B.1. *Let Z be a random variable that takes values in $[0,1]$. Assume that $\mathbb{E}[Z] = \mu$. Then, for any $a \in (0,1)$,*

$$\mathbb{P}[Z > 1-a] \geq \frac{\mu - (1-a)}{a}.$$

Apply the lemma for $Z = L_{\mathcal{D}}(A'(S))$, $a = 7/8$, $\mu = E[Z] \geq 1/4$. We obtain:

$$P(Z \geq 1-\frac{7}{8}) \geq \frac{\frac{1}{4} - (1-\frac{7}{8})}{\frac{7}{8}} \Leftrightarrow P(Z \geq \frac{1}{8}) \geq \frac{\frac{1}{4} - \frac{1}{8}}{\frac{7}{8}} = \frac{\frac{1}{8}}{\frac{7}{8}} = \frac{1}{7}$$

$$\mathbb{P}[L_{\mathcal{D}}(A'(S)) \geq 1/8] \geq 1/7$$

Lecture 3: Universal concept class U_n

- B^n = set of boolean n-tuples, $|B| = 2^n$
- want to learn arbitrary subsets of B^n
- $\mathcal{U}_n = \{h: B^n \rightarrow \{0,1\}\}$ - the concept class formed by all subsets of B^n
- $\mathcal{U}_n$ – universal class
- $|\mathcal{U}_n| = 2^{2^n}$ – finite, so is PAC learnable with $m_{\mathcal{H}}(\varepsilon, \delta)$ in the order of m:

$$m \geq \left\lceil \frac{1}{\varepsilon}\left(2^n \log(2) + \log(\frac{1}{\delta}) \right) \right\rceil$$

- $\mathcal{U}_n$ – finite class, from the No-Free-Lunch theorem we need to see $m > 2^{n-1}$ training examples otherwise there exists a task $(f, \mathcal{D})$ on which the learner will fail

$$P_{S \sim D^m}(L_D(A(S)) \geq \frac{1}{8}) \geq \frac{1}{7}$$

- for $\varepsilon = 1/8$, $\delta = 6/7$, $n = 10$, $m \geq 5679$ (all examples are $2^{10} = 1024$), no matter how $\mathcal{D}$ looks like we have:

$$P_{S \sim D^m}(L_D(A(S)) \geq \frac{1}{8}) \leq \frac{1}{7}$$

No-Free-Lunch and prior knowledge

Consider $\mathcal{H}$ the set of all functions from $\mathcal{X}$ to $\mathcal{Y}$, $\mathcal{H} = \{h: \mathcal{X} \to \mathcal{Y}\}$. This class represents lack of prior knowledge: every member of it is a good candidate.

According to the No-Free-Lunch theorem, any learning algorithm (in particular the ERM predictor) that chooses its output from hypothesis in $\mathcal{H}$ will fail on some learning task. Therefore, $\mathcal{H}$ is not PAC learnable.

Corollary

Let $\mathcal{X}$ be an infinite domain set and let $\mathcal{H}$ be the set of all functions from $\mathcal{X}$ to $\{0, 1\}$. Then $\mathcal{H}$ is not PAC learnable.

Proof: Choose $\varepsilon < 1/8$, $\delta < 6/7$ and then apply the No-Free-Lunch theorem. There exists an f in $\mathcal{H}$ and a distribution $\mathcal{D}$ such that $L_{\mathcal{D}}(f) = 0$, but no matter how many examples a learning algorithm A will receives it will fail on the task as $P(L_{\mathcal{D}}(A(S)) \geq 1/8) \geq 1/7$ (contradiction with PAC learnability).

No-Free-Lunch and prior knowledge

We can escape the hazards foreseen by the No-Free-Lunch theorem by using our prior knowledge about a specific learning task, to avoid the distributions that will cause us to fail when learning that task.

Such prior knowledge can be expressed by restricting our hypothesis class. But how should we choose a good hypothesis class?

On the one hand, we want to believe that this class includes the hypothesis that has no error at all (in the PAC setting), or at least that the smallest error achievable by a hypothesis from this class is indeed rather small (in the agnostic setting).

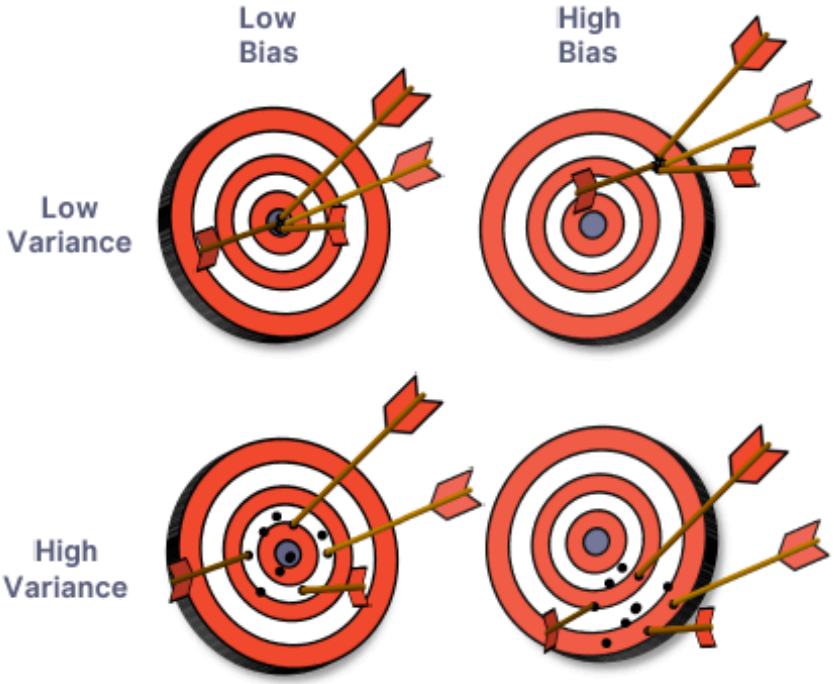
We cannot simply choose the richest class – the class of all functions over the given domain. Discuss this tradeoff next.

The Bias-Complexity tradeoff

Bias vs. Variance trade-off

Bias vs. Variance

- There are two sources of error in ML models.
- **Bias** (or **systematic error**) comes from the inability of the algorithm to model the true relationship between features and label (*underfitting*).
  - Bias error cannot be corrected by adding more training samples, but by increasing the complexity of the model.
- **Variance** (or **random error**) comes from sensitivity to small fluctuations in the training data, causing the algorithm to model random noise. (*overfitting*)
  - Variance error can be corrected by adding more training samples or by decreasing the complexity of the model.
- There is usually a *tradeoff* between bias and variance.



Slide credit: Sparktech

The error decomposition

Decompose the error of an $\mathrm{ERM}_{\mathcal{H}}$ predictor that chooses h_S from a restricted class $\mathcal{H}$ into two components:

$$L_{\mathcal{D}}(h_S) = \epsilon_{\text{app}} + \epsilon_{\text{est}}$$

$$\text{where}: \quad \epsilon_{\text{app}} = \min_{h \in \mathcal{H}} L_{\mathcal{D}}(h), \quad \epsilon_{\text{est}} = L_{\mathcal{D}}(h_S) - \epsilon_{\text{app}}$$

The approximation error (ε_{app})

- the minimum risk achievable by a predictor in the hypothesis class $\mathcal{H}$
- it measures how well our hypothesis class $\mathcal{H}$ fits the distribution
- it is independent of the particular sample
- it is determined only by $\mathcal{H}$, enlarging it decreases the approximation error
- under the realizability assumption, the approximation error is zero.
- it is the ***bias*** term

The error decomposition

Decompose the error of an $\mathrm{ERM}_{\mathcal{H}}$ predictor that chooses h_S from a restricted class $\mathcal{H}$ into two components:

$$L_{\mathcal{D}}(h_S) = \epsilon_{\text{app}} + \epsilon_{\text{est}}$$

$$\text{where}: \quad \epsilon_{\text{app}} = \min_{h \in \mathcal{H}} L_{\mathcal{D}}(h), \quad \epsilon_{\text{est}} = L_{\mathcal{D}}(h_S) - \epsilon_{\text{app}}$$

The estimation error (ε_{est})

- it measures how well our particular sample let us estimate the best classifier
- the empirical risk is only an estimate of the true risk, and so the predictor minimizing the empirical risk is only an estimate of the predictor minimizing the true risk.
- the quality of this estimation depends on the training set size and on the size (complexity) of $\mathcal{H}$
- it varies with samples
- it is the ***variance/complexity*** term

The bias – complexity tradeoff

On one hand, choosing $\mathcal{H}$ to be a very rich class decreases the approximation error but at the same time might increase the estimation error, as a rich $\mathcal{H}$ might lead to overfitting .

On the other hand, choosing $\mathcal{H}$ to be a very small set reduces the estimation error but might increase the approximation error or, in other words, might lead to underfitting .

A great choice for $\mathcal{H}$ is the class that contains only one classifier – the Bayes optimal classifier. But the Bayes optimal classifier depends on the underlying distribution $\mathcal{D}$, which we do not know (indeed, learning would have been unnecessary had we known $\mathcal{D}$).

Learnability - what do we know so far?

Learnability - what do we know so far?

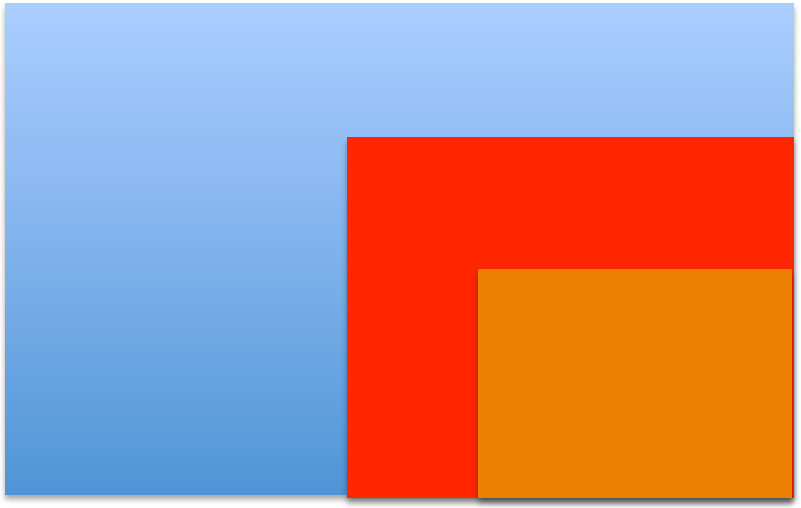
Let $\mathcal{H}$ a hypothesis class.

- is it PAC learnable?
- is it agnostic PAC learnable?
  - we do know that agnostic PAC learnability → PAC learnability

all hypothesis classes

PAC learnable

agnostic PAC learnable



Definition 3.1 (PAC Learnability). A hypothesis class $\mathcal{H}$ is PAC learnable if there exist a function $m_{\mathcal{H}} : (0,1)^2 \to \mathbb{N}$ and a learning algorithm with the following property: For every $\epsilon, \delta \in (0,1)$, for every distribution $\mathcal{D}$ over $\mathcal{X}$, and for every labeling function $f : \mathcal{X} \to \{0,1\}$, if the realizable assumption holds with respect to $\mathcal{H}, \mathcal{D}, f$, then when running the learning algorithm on $m \geq m_{\mathcal{H}}(\epsilon, \delta)$ i.i.d. examples generated by $\mathcal{D}$ and labeled by f, the algorithm returns a hypothesis h such that, with probability of at least $1 - \delta$ (over the choice of the examples), $L_{(\mathcal{D},f)}(h) \leq \epsilon$.

Definition 3.4 (Agnostic PAC Learnability for General Loss Functions). A hypothesis class $\mathcal{H}$ is agnostic PAC learnable with respect to a set Z and a loss function $\ell : \mathcal{H} \times Z \to \mathbb{R}_+$, if there exist a function $m_{\mathcal{H}} : (0,1)^2 \to \mathbb{N}$ and a learning algorithm with the following property: For every $\epsilon, \delta \in (0,1)$ and for every distribution $\mathcal{D}$ over Z, when running the learning algorithm on $m \geq m_{\mathcal{H}}(\epsilon, \delta)$ i.i.d. examples generated by $\mathcal{D}$, the algorithm returns $h \in \mathcal{H}$ such that, with probability of at least $1 - \delta$ (over the choice of the m training examples),

$$L_{\mathcal{D}}(h) \leq \min_{h' \in \mathcal{H}} L_{\mathcal{D}}(h') + \epsilon,$$

where $L_{\mathcal{D}}(h) = \mathbb{E}_{z \sim \mathcal{D}} [\ell(h, z)]$.

Learnability - what do we know so far?

Let $\mathcal{H}$ a hypothesis class.

- is it PAC learnable?
- is it agnostic PAC learnable?
  - we do know that agnostic PAC learnability → PAC learnability
  - we don't know that PAC learnability → agnostic PAC learnability

Size of class $\mathcal{H}$:

- finite
- infinite

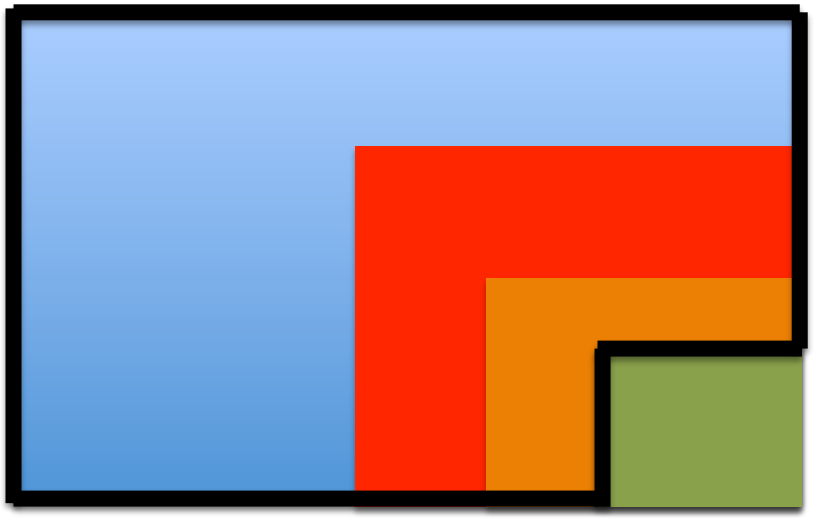
all hypothesis classes

PAC learnable

agnostic PAC learnable

finite size classes

infinite size classes



Learnability - what do we know so far?

Let $\mathcal{H}$ a hypothesis class.

- is it PAC learnable?
- is it agnostic PAC learnable?
  - we do know that agnostic PAC learnability → PAC learnability
  - we don't know that PAC learnability → agnostic PAC learnability

Size of class $\mathcal{H}$:

- finite
- infinite

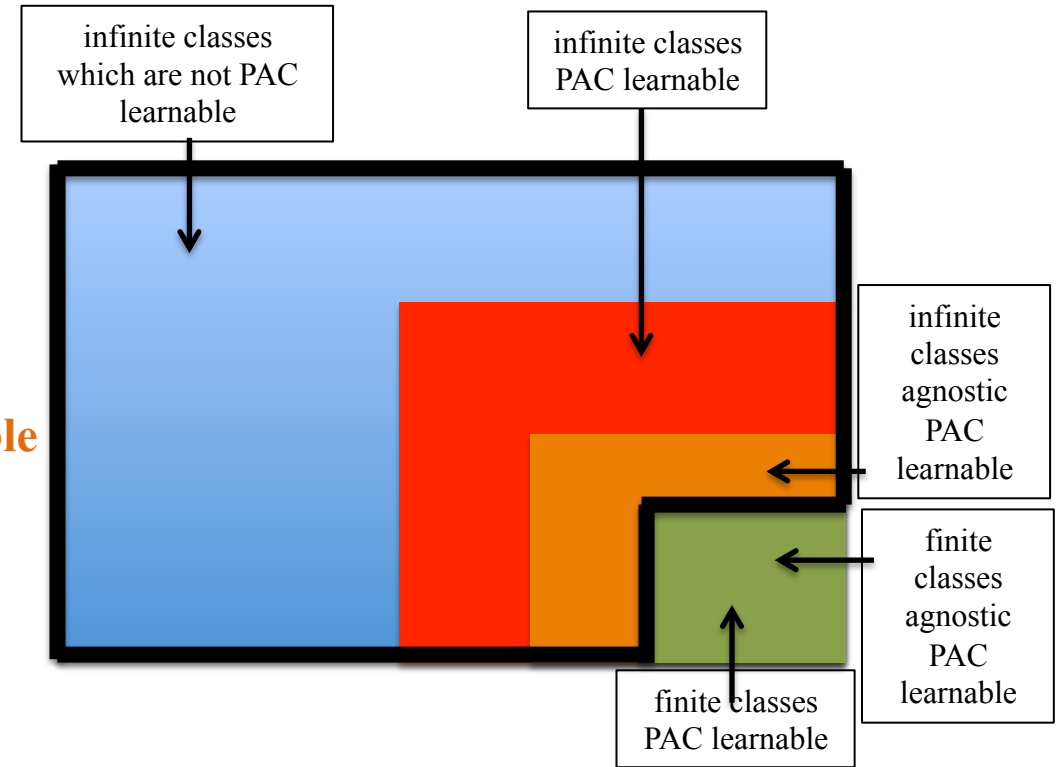
all hypothesis classes

PAC learnable

agnostic PAC learnable

finite size classes

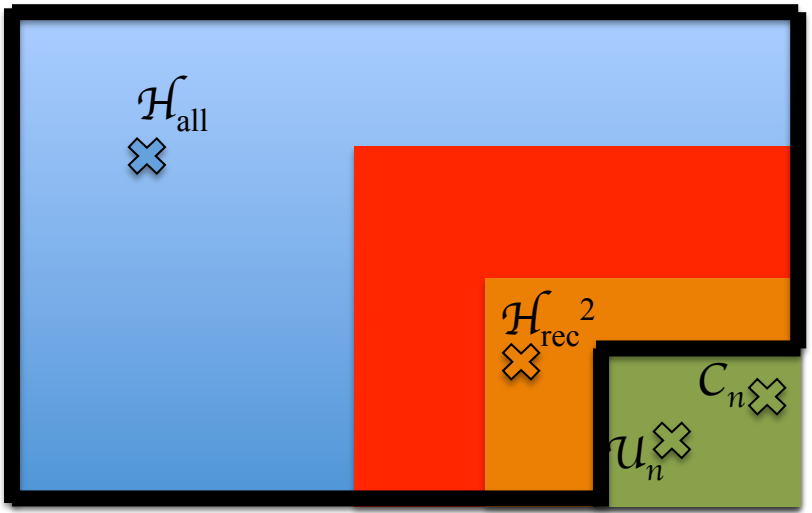
infinite size classes



Learnability - what do we know so far?

Hypothesis classes $\mathcal{H}$ encountered until now:

- finite $\mathcal{H}$
  - C_n concept class of conjunctions of at most n Boolean literals $x_1, \ldots, x_n$
  - $\mathcal{U}_n$ universal concept class
- infinite $\mathcal{H}$
  - $\mathcal{H}_{rec}^2$ set of all axis-aligned rectangle lying in R^2 (with positive labels unaffected – PAC or affected by noise – agnostic PAC)
  - $\mathcal{H}_{all}$ all functions from X to {0,1}(No Free-Lunch theorem)



all hypothesis classes

PAC learnable

agnostic PAC learnable

finite size classes

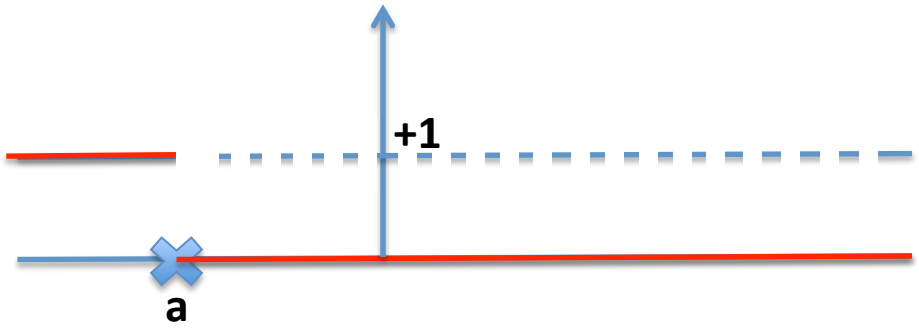
infinite size classes

Another class example - $\mathcal{H}_{\text{thresholds}}$

Consider $\mathcal{H}_{\text{thresholds}}$ be the set of threshold functions over the real line

$\mathcal{H}_{\text{thresholds}} = \{h_a : \mathrm{R} \rightarrow \{0, 1\}, h_a(\mathrm{x}) = \mathbf{1}_{[x<a]}, a \in \mathbf{R}\}, |\mathcal{H}_{\text{thresholds}}| = \infty$

$$\mathbf{1}_{[x<a]} = \begin{cases} 1, & x < a \\ 0, & x \geq a \end{cases},$$



is the indicator function of the set $\{x \in \mathbf{R} \mid x < a\}$

Lemma

$\mathcal{H}_{\text{thresholds}}$ is PAC learnable, using the ERM learning rule, with sample complexity of

$$m_{H_{thresholds}} \leq \left\lceil \frac{1}{\varepsilon} \log \frac{1}{\delta} \right\rceil$$

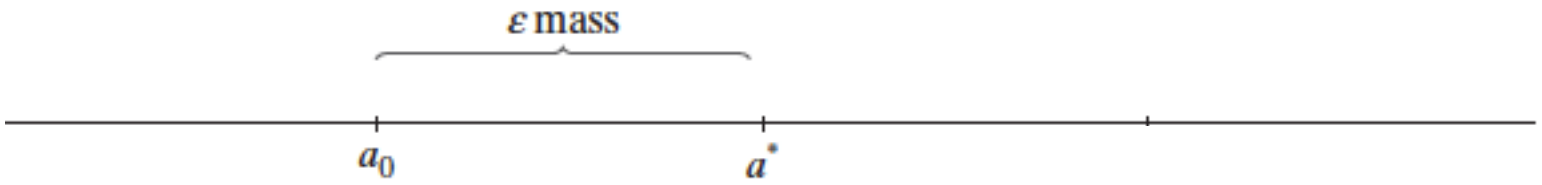
Another class example - $\mathcal{H}_{\text{thresholds}}$

Proof

Let $a^* \in \mathbf{R}$ such that $h_{a^*}(x) = \mathbf{1}_{[x<a^*]}$ achieves $L(h_{a^*}) = 0$ (realizability assumption in the PAC learning scenario).

Consider $\mathcal{D}_x$ a distribution over $\mathcal{X} = \mathbf{R}$ and take $a_0 < a^* \in \mathbf{R}$ such that:

$$\mathbb{P}_{x\sim\mathcal{D}_x}[x \in (a_0, a^\star)] = \epsilon.$$



If $D_x(-\infty, a^*) \leq \varepsilon$ take $a_0 = -\infty$.

Consider the training set $S = ((x_1,y_1), \ldots, (x_m,y_m))$ and the following algorithm A:

- take $b = \max\{x_i | (x_i, 1) \in S\}$ (if no positive example appears in S take $b = -\infty$)
- output $A(S) = h_S = h_b$

Another class example - $\mathcal{H}_{\text{thresholds}}$

Then $L_D(h_S) > \varepsilon$ means that $b < a_0$. We have that:

$$\underset{S\sim D^m}{P}(L_D(h_S) > \varepsilon) = P(b < a_0) = (1-\varepsilon)^m \leq e^{-\varepsilon m}$$

Take $$m = \left\lceil \frac{1}{\varepsilon} \log \frac{1}{\delta} \right\rceil$$

So, we have that:

$$\underset{S\sim D^m}{P}(L_D(h_S) > \varepsilon) < \delta$$

which shows that $\mathcal{H}_{\text{thresholds}}$ is PAC learnable, using the ERM learning rule.

No Free Lunches vs. $\mathcal{H}_{\text{thresholds}}$

Why is $\mathcal{H}_{\text{thresholds}}$ = set of threshold classifiers not a victim of the No Free Lunch theorem? (we can PAC learn them)

The reason is simple:

- the class of threshold classifiers is so simple that an adversary has no room to create an adversarial distribution

In fact, as our discussion above shows:

- if two threshold classifiers agree on a large enough sample
- their respective thresholds will be close to each other
- there is no way you can force them to behave completely differently on unseen examples.

If that would have been possible then:

- we would have been able to create an adversarial distribution.

So, it seems necessary for PAC learnability that the general class $\mathcal{H}$ considered isn't too expressive