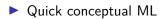
## PAC Intro from Understanding ML

Cody Melcher

February 27, 2023

◆□▶ ◆□▶ ◆ 臣▶ ◆ 臣▶ ○ 臣 ○ の Q @





1. Necessary for complex tasks, useful for inherent flexibility.

(ロ)、(型)、(E)、(E)、(E)、(O)へ(C)

#### Overview

Quick conceptual ML

1. Necessary for complex tasks, useful for inherent flexibility.

▲□▶ ▲□▶ ▲ 三▶ ▲ 三▶ 三三 - のへぐ

- Learning Environment
  - 1. Framework
  - 2. ERM (without and with Inductive Bias)

#### Overview

Quick conceptual ML

1. Necessary for complex tasks, useful for inherent flexibility.

▲□▶ ▲□▶ ▲ 三▶ ▲ 三▶ 三 のへぐ

Learning Environment

1. Framework

2. ERM (without and with Inductive Bias)

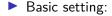
Formal Definition of the PAC learning model

1. General bounds

# ML Conceptually

- ML: Automating converting "experience" or data into knowledge.
  - 1. Necessary for complex tasks, useful for inherent flexibility.

## Statistical Learning Framework



- 1. **Domain space**  $\mathcal{X}$  ie covariate space.
- 2. Label set: space of possible responses/explanatory variables. Focus here on  $\mathcal{Y}=\{1,2\}$
- 3. Training data as finite sequence of ordered pairs in  $\mathcal{X} \times \mathcal{Y}$
- 4. **Prediction rule**  $h : \mathcal{X} \to \mathcal{Y}$ ; also called hypothesis or classifier.

▲□▶ ▲□▶ ▲□▶ ▲□▶ ■ ●の00

## Statistical Learning Framework continued

- Notes on basic framework
  - 1. Assume data generated from unknown distribution  $\ensuremath{\mathcal{D}}$  Generally assume iid.
  - Assume perfect/correct but unknown classifier f<sub>i</sub> exists and y<sub>i</sub> are mapped to by f<sub>i</sub> from X.
  - 3. Error of h is

 $L_{\mathcal{D},\{}(h) = P_{x \sim \mathcal{D}}(h(x) \neq f(x) = \mathcal{D}(\{x : h(x) \neq f(x)\}).$ Minimize!

▲□▶ ▲□▶ ▲□▶ ▲□▶ ■ ●の00

- 4. Note  $\mathcal{D} : A \rightarrow [0, 1]$  where  $A \subset \mathcal{X}$
- 5. L called risk or generalization error

## Statistical Learning Framework continued

#### Empirical Risk Minimization

Don't know D or f. Minimize training error (empirical risk/error) instead.

2. 
$$L_S(h) = \frac{|\{i \in [m]: h(x_i) \neq y_i\}|}{m} \in [0, 1]$$

- Problem: can easily lead to overfitting ie define perfect classifier for training data that is terrible classifier on other data.
- Solution: restriction on set of possible classifiers *H*. Big topic: conditions on *H* to guarantee no overfitting.
- 5. First restriction:  $|\mathcal{H}| < \infty$  which leads to...

#### **ERM**

- Realizability Assumption (2.1) etc.
  - 1. There exists  $h^* \in \mathcal{H}$  st  $L_{D,f}(h^*) = 0$  ie there exists for every empirical risk minimization problem there exists a perfect classifier.
  - Possible issue: data drawn from D could be bad representation of D. Need to think of error as random.
  - Let δ be probability sample from D is non-representative sample so (1 – δ) is our confidence parameter (our sample is representative).
  - 4. Let  $\epsilon$  be the accuracy parameter
  - 5. Want to use these to put upper bound on what samples can be realized that lead to classification failure.

・ロト・西ト・ヨト・ヨト・ 日・ うらぐ

- Want to figure out bounding D<sup>m</sup>({S|<sub>x</sub> : L<sub>(D,f)</sub>(h<sub>s</sub>) > ε}) le bound the realized samples of size m mapped by the classifier incorrectly classifies by ε.
- Define the set of bad classifiers as

$$\mathcal{H}_B = \{h \in \mathcal{H} : L_{D,f}(h) > \epsilon\}$$
 and let

 $M = \{S|_{x} : \exists h \in \mathcal{H}_{B}, L_{S}(h) = 0\}$  be the set of misleading

samples.

Follows that {S<sub>x</sub> : L<sub>D,f</sub>(h<sub>s</sub>) > ε} ⊂ M is set of samples that give a bad classifier are a subset of the misleading samples set.

- Can relatedly think of all the possible bad classifiers that manage to give perfect prediction due to a misleading sample; this is equivalent to M.
- It follows that

 $D^{m}(\{S|_{x}: L_{(D,f)}(h_{s}) > \epsilon\}) \leq D^{m}(\cup_{h \in H_{b}}\{S|_{x}: L_{s}(h) = 0\})$ 

- ► By  $D(A \cup B) \le D(A) + D(B)$ , it follows that RHS  $\le \sum_{h \in H_b} D^m(\{S|_x : L_S(h) = 0\})$
- Next due to iid assumption and how we defined  $\epsilon$ , it follows that  $D^m(\{x_i : h(x_i) = y_i\}) = (1 - L_{D,f})^m \le (1 - \epsilon)^m \le e^{-\epsilon m}$

- Combining the previous inequality with the inequality from two slides back we get, D<sup>m</sup>({S|<sub>x</sub> : L<sub>(D,f)</sub>(h<sub>s</sub>) > ε}) ≤ |H|e<sup>-εm</sup> which provides the upper bound on the realized samples of size m mapped by the classifier incorrectly, which we can summarize as,
- Corollary 2.3: If H is finite, δ ∈ (0, 1) and ε > 0 and let m be any integer st. m ≥ log((|H|/δ))<sup>1</sup>/ε. Then for any true classifier f and distribution D assuming realizability wit probability at least 1 − δ over iid sample of size m, we have that for every ERM classifier h<sub>s</sub> it holds that L<sub>D,f</sub>(h<sub>s</sub>) ≤ ε.

► le. for sufficiently large m, classifiers from finite classifier class will be probably (with confidence 1 − δ) approximately (up to ε wrong) correct.

▲□▶ ▲□▶ ▲□▶ ▲□▶ ■ ●の00

Hence why, PAC learning environment = probably approximately correct.

#### PAC Learnability Definition

▶ (3.1) PAC Learnability: H is PAC learnable if there exists a function  $m_H: (0,1)^2 \to \mathcal{N}$  and a learning algorithm with the following property:  $\forall \epsilon, \delta \in (0, 1), \forall \mathcal{D} \text{ over } \mathcal{X} \text{ and for every}$ true classifier  $f : \mathcal{X} \to 0, 1$ , and if the realizable assumptions holds, then when running the learning algorithm on  $m \geq m_h(\epsilon, \delta)$  iid samples generated by  $\mathcal{D}$  and classified by f, the algorithm returns a classifier h st with probability at least  $1-\delta$  that  $L_{D,f}(h) < \epsilon$ 

#### PAC Learnability notes

- $\blacktriangleright \epsilon$  = how far the classifier can be from the optimal classifier f
- $\delta$  how likely the classifier is to be  $\epsilon$  close to f
- notice that m<sub>h</sub> determines how many samples needed to be probably accurate; we will focus on finding the minimal m<sub>h</sub> ie minimal integer that guarantees probable accuracy.
  log(|H|)

$$\blacktriangleright (3.2) \ \forall |\mathcal{H}| < \infty, \ m_h(\epsilon, \delta) \le \frac{\log(\frac{|\mathcal{H}|}{\delta})}{\epsilon}$$

### Agnostic PAC

- Realizability seems unrealistic....solution: agnostic PAC!
- (3.3) A hypothesis class *H* is agnostic PAC learning if
   ∃m<sub>h</sub>: (0,1)<sup>2</sup> → *N* and a learning algorithm with the following property: ∀ε, δ ∈ (0,1) and ∀ *D* over *X* × *Y* when running the learning algorithm on m ≥ m<sub>h</sub>(ε, δ) iid examples generated by *D*, a hypothesis h is returned st with probability at least 1 − δ, L<sub>D</sub>(h) ≤ min<sub>h'∈H</sub>(L<sub>D</sub>(h') + ε)

#### Extensions to Agnostic PAC

- Generally we refer to the agnostic PAC learning environment as PAC learning.
- Can expand *Y* to be larger than {0,1}, but still assuming it is finite.
- Can change risk to be expected squared difference (or something else) to deal with regression problems.
- ▶ Both these extensions change  $L_D(h)$ , and thus we need to rethink our above definition....

▲□▶ ▲□▶ ▲□▶ ▲□▶ ■ ●の00

#### Agnostic PAC with General Loss

- (3.4) A hypothesis class  $\mathcal{H}$  is agnostic PAC learning with respect to set Z and loss function  $I: H \times Z \rightarrow \mathcal{R}_+$  if  $\exists m_h : (0,1)^2 
  ightarrow \mathcal{N}$  and a learning algorithm with the following property:  $\forall \epsilon, \delta \in (0, 1)$  and  $\forall \mathcal{D}$  over  $\mathcal{Z}$  when running the learning algorithm on  $m \geq m_h(\epsilon, \delta)$  iid examples generated by  $\mathcal{D}$ , a hypothesis h is returned st with probability at least  $1-\delta$ ,  $L_D(h) \leq \min_{h' \in H} (L_D(h') + \epsilon)$
- ▶ where L<sub>D</sub>(h) = E<sub>(z∼D)</sub>(I(h, z) and Z = X × Y (for our problems though this can be generalized)

## TLDR;

- PAC learning feels natural to stats/math world, learning from "experience" ie data ie we are still trying to approximate a function.
- New ish to us maybe because PAC learning is about thinking about bounds on what is possible and sorta avoiding asymptotics.
- Still very general, but allows us to move to VC dimension (what characteristics of *H* allow us to deduce it is PAC learnable?), be more rigorous with a lot of the techniques we have learned, etc.