Machine learning theory

Probably approximately correct model

Hamid Beigy

Sharif University of Technology

February 25, 2023

Table of contents

Introduction

1. The consistency model is not a particularly great model of learning, but it's simple and good to start.

Definition (Consistency model)

We say that algorithm $\mathcal{A}$ learns the concept class $\mathcal{C}$ in the consistency model if given any training set S, the algorithm produces a hypothesis (concept) $c \in \mathcal{C}$ consistent with S **if one exists**, and outputs **"there is no consistent concept"** otherwise.

Definition (Learnability of consistency model)

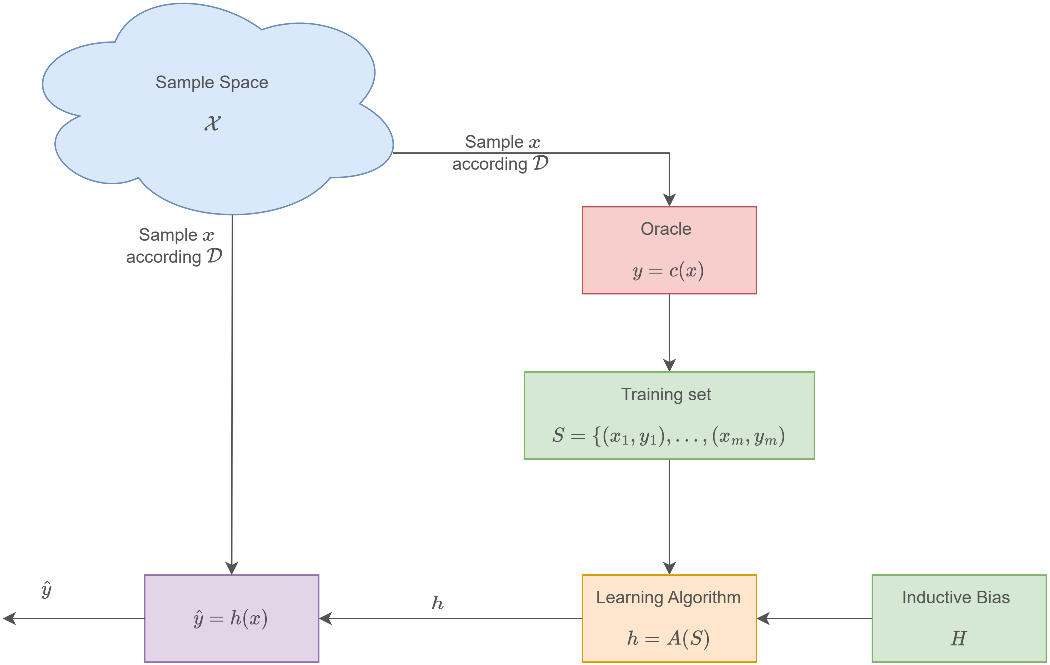
We say that a class $\mathcal{C}$ is **learnable** in the consistency model **if there exists an efficient algorithm** $\mathcal{A}$ that learns $\mathcal{C}$ in the consistency model.

1. Here **efficient** means that **the algorithm runs in polynomial time** in terms of the size of the set S and the size of each $x \in S$.
2. The examples given in the previous lecture showed a few shortcomings of the consistency model.
   - A class $\mathcal{C}$ can be learnable while a subclass of $\mathcal{C}$ can be unlearnable.
   - The consistency model yields a concept that tells us nothing about the accuracy of the model on new data (generalization).
   - The consistency model has a practical problem in that training data that contains noise is not handled in a robust way.

1. We need to add the distribution from which training and test examples are generated.
2. We assume that the following conditions hold.
   - Training and test examples are generated from some unknown distribution $\mathcal{D}$.
   - Each example is generated independently.
   - There exists a function $c \in \mathcal{C}$ (concept) such that each example labeled according to c.
   - We would like the results to be distribution-free (the results hold for any target distribution).
3. Learning algorithm receives training examples and outputs a hypothesis $h \in H$..
4. Hypothesis h makes a mistake if $h(x) \neq c(x)$.
5. We measure error of learning algorithm using generalization error.

$$\mathbf{R}(h) = \underset{x \sim \mathcal{D}}{\mathbb{P}} [h(x) \neq c(x)]$$

6. We aim to have $\mathbf{R}(h)$ be **small**.


Sample Space
$\mathcal{X}$
Sample x according $\mathcal{D}$
Sample x according $\mathcal{D}$
Oracle
$y = c(x)$
Training set
$S = \{(x_1, y_1), \dots, (x_m, y_m)\}$
$\hat{y}$
$\hat{y} = h(x)$
h
Learning Algorithm
$h = A(S)$
Inductive Bias
H

1. We aim to have $\mathbf{R}(h)$ be **small**.

Definition (Approximately correct hypothesis)

Hypothesis h is called approximately correct if $\mathbf{R}(h) \leq \epsilon$ (for small ϵ).

2. Parameter ϵ is called accuracy parameter.
3. We can't always guarantee that $\mathbf{R}(h) \leq \epsilon$ because, depending on training set, the training data may be a very unrepresentative of the domain set.
4. We require that we are able to learn **a good approximation** with **high probability**.
5. In particular, we require that $\mathbf{R}(h) \leq \epsilon$ with probability at least $1 - \delta$.
6. This hypothesis is called **probably approximately correct**.
7. Parameter δ is called confidence parameter.

Probably approximately correct model

Definition (PAC Learnability)

A concept class $\mathcal{C}$ is **PAC-learnable** by hypothesis class H if there exists an algorithm $\mathcal{A}$, such that for all target concepts $c \in \mathcal{C}$, for all distributions $\mathcal{D}$ on $\mathcal{X}$, for all $\epsilon, \delta \in (0, 1)$, the algorithm $\mathcal{A}$ takes

$$m \geq m_H(\epsilon, \delta) = poly\left(\frac{1}{\epsilon}, \frac{1}{\delta}, n, |c|\right)$$

examples in form of $S = \{(\mathbf{x}_1, c(\mathbf{x}_1)), \ldots, (\mathbf{x}_m, c(\mathbf{x}_m))\}$, where each $\mathbf{x}_i$ is chosen from space $\mathcal{X}$ at random according to the target distribution $\mathcal{D}$, and produces a hypothesis $h \in H$, such that

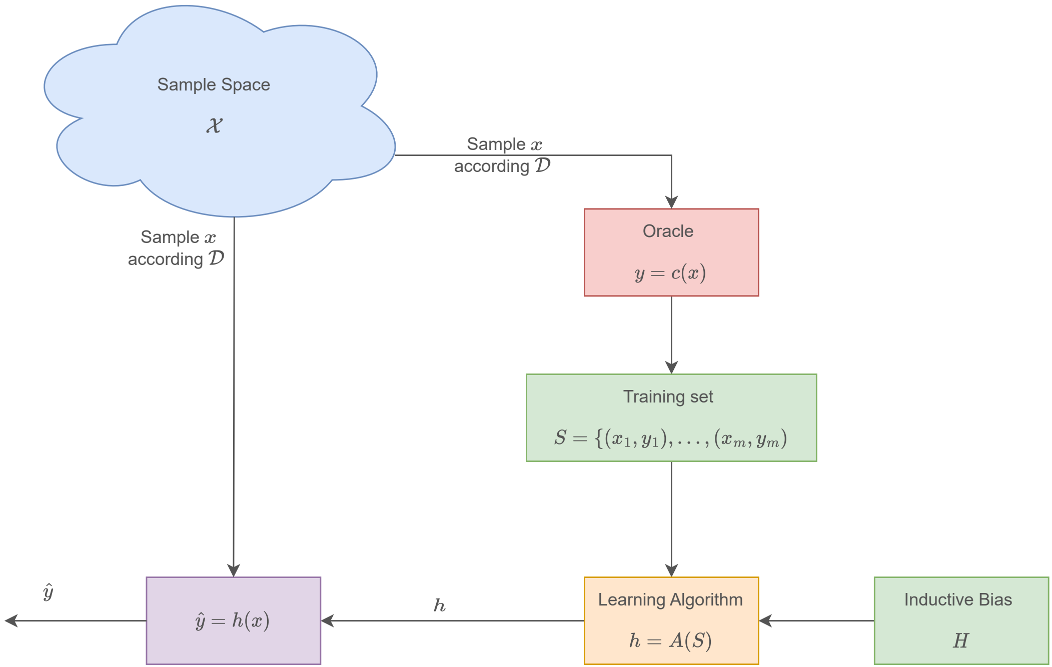
$$\mathbb{P}_{S \sim \mathcal{D}^m}\left[\mathbf{R}(h) \leq \epsilon\right] \geq (1 - \delta)$$

Definition (Sample complexity)

Function $m_H : (0,1)^2 \mapsto \mathbb{N}$ that measures how many samples are required to guarantee PAC learnability of H, is called **sample complexity**.

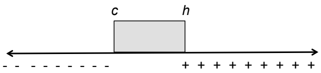
Definition (Efficiently PAC-learnable)

A concept class $\mathcal{C}$ is **efficiently PAC-learnable** by hypothesis class H if there exists an algorithm $\mathcal{A}$ that runs in *poly* $\left(\frac{1}{\epsilon}, \frac{1}{\delta}, n, |c|\right)$ time and PAC-learns $\mathcal{C}$ using H.

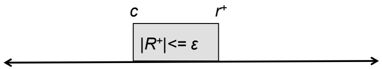

Sample Space
$\mathcal{X}$
Sample x according $\mathcal{D}$
Oracle
$y = c(x)$
Sample x according $\mathcal{D}$
Training set
$S = \{(x_1, y_1), \ldots, (x_m, y_m)\}$
$\hat{y}$
$\hat{y} = h(x)$
h
Learning Algorithm
$h = A(S)$
Inductive Bias
H

Example (Learning the threshold function)

1. Let $\mathcal{X} = \mathbb{R}$ and $\mathcal{C} = \{positive\ half\ lines\}$.
2. For some point c, the corresponding positive half line is the region of $\mathbb{R}$ designated by $[c, \infty)$.
3. We can treat $c \in \mathcal{C}$ as a point that separates the positive and negative regions of $\mathbb{R}$.



4. There will be a region between c and h in which h will incorrectly label new training points.
5. We want to find a h such that this region have size $\leq \epsilon$ in terms of true distribution $\mathcal{D}$.
6. We have two bad cases:
   - B^+ h lies more than ϵ to the right of c.
   - B^- h lies more than ϵ to the left of c.
7. First considering the likelihood of B^+.



8. Let R^+ be the smallest region with c as its left border whose probability mass is at least ϵ.
9. That is $R^+ = [c, r^+]$, where $r^+ = sup\{r \geq c \mid \mathbb{P}[[c, r)] \leq \epsilon\}$.

Example (Learning the threshold function (cont.))

1. Let $\mathcal{A}$ be learning algorithm that returns the value of the smallest positive example.
2. Hence, h falls to the right of r^+ only if all training examples lie outside of R^+.
3. Therefore, B^+ only occurs when no training points fall in R^+.
4. Thus, if R^+ has size ϵ, then $\mathbb{P}\left[x_1 \notin R^+\right] \leq (1-\epsilon)$.
5. Given m training examples, we have

$$\begin{aligned}\mathbb{P}\left[B^+\right] &= \mathbb{P}\left[(x_1 \notin R^+) \wedge \ldots \wedge (x_m \notin R^+)\right] \\ &= \mathbb{P}\left[x_1 \notin R^+\right] \times \ldots \times \mathbb{P}\left[x_m \notin R^+\right] \leq (1-\epsilon)^m\end{aligned}$$

6. We have bad cases, hence

$$\begin{aligned}\mathbb{P}\left[\mathbf{R}(h) > \epsilon\right] &= \mathbb{P}\left[B^+ \vee B^-\right] \\ &\leq \mathbb{P}\left[B^+\right] + \mathbb{P}\left[B^-\right] \\ &\leq 2(1-\epsilon)^m \\ &\leq 2e^{-\epsilon m} \leq \delta. && \textit{using } 1+x \leq e^x\end{aligned}$$

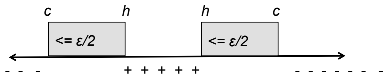
7. Then $m \geq \frac{1}{\epsilon}\ln\left(\frac{2}{\delta}\right)$.
8. Rearranging terms, we see what with probability of at least $(1-\delta)$, we have $\mathbf{R}(h) \leq \frac{1}{m}\ln\left(\frac{2}{\delta}\right)$.

Example (Learning an interval)

1. Let $\mathcal{X} = \mathbb{R}$ and $\mathcal{C} = \{intervals\}$. In this concept class, we have

$$c(x) = \begin{cases} 1 & \text{if } x \in [a, b] \\ 0 & \text{otherwise} \end{cases}$$

2. We have 2 boundary error regions. We force size of each region at most $\frac{\epsilon}{2}$.

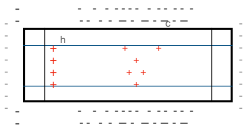


3. Hence, we have 4 possible bad events.
4. A similar analysis as for the previous example, we have sample complexity of

$$m_H(\epsilon, \delta) \geq \frac{2}{\epsilon} \ln\left(\frac{4}{\delta}\right)$$

Example (Learning axis-aligned rectangles)

1. Let $\mathcal{X} = \mathbb{R}^2$ and $\mathcal{C} = \{axis - aligned\ rectangles\}$. In this concept class, we have



$$c(\mathbf{x}) = \begin{cases} 1 & \text{if } \mathbf{x} \text{ belongs to the given rectangle} \\ 0 & \text{otherwise} \end{cases}$$

2. We have 4 boundary error regions. We force size of each region at most $\frac{\epsilon}{4}$.
3. A similar analysis as for the previous example, we have sample complexity of

$$m_H(\epsilon, \delta) \geq \frac{4}{\epsilon} \ln\left(\frac{4}{\delta}\right)$$

Example (Learning axis-aligned hyper-rectangles)

Find the sample complexity of $\mathcal{X} = \mathbb{R}^n$ and $\mathcal{C} = \{axis - aligned\ hyper - rectangles\}$.

Learning bound for finite H

Theorem (Learning bound for finite H)

Let $H = \{h \mid h : \mathcal{X} \mapsto \{0,1\}\}$ be a finite set of functions and $\mathcal{A}$ an algorithm that for any target concept $c \in H$ and sample S, returns a consistent hypothesis $h \in H$. Then, for any $\delta > 0$, with probability at least $(1-\delta)$, we have

$$m_H(\epsilon,\delta) \geq \frac{1}{\epsilon}\left(\log|H| + \log\left(\frac{1}{\delta}\right)\right)$$
$$\mathbf{R}(h) \leq \frac{1}{m}\left(\log|H| + \log\left(\frac{1}{\delta}\right)\right)$$

Proof (Learning bound for finite H).

For any $\epsilon > 0$, define $H_\epsilon = \{h \in H \mid \mathbf{R}(h) > \epsilon\}$. Then,

$$\begin{aligned}
\mathbb{P}\left[\exists h \in H_\epsilon \,\middle|\, \hat{\mathbf{R}}(h) = 0\right] &= \mathbb{P}\left[(\hat{\mathbf{R}}(h_1) = 0) \vee (\hat{\mathbf{R}}(h_2) = 0) \vee \ldots \vee (\hat{\mathbf{R}}(h_{|H_\epsilon|}) = 0\right] \\
&\leq \sum_{h_i \in H_\epsilon} \mathbb{P}\left[\hat{\mathbf{R}}(h_i) = 0\right] \leq \sum_{h_i \in H_\epsilon} (1-\epsilon)^m \\
&\leq |H|(1-\epsilon)^m \leq |H|e^{-m\epsilon} \leq \delta
\end{aligned}$$

□

Corollary

This theorem shows that when hypothesis space H is finite,

1. *a consistent algorithm A is a PAC-learning algorithm,*
2. *specifies an upper bound on how much data we need to achieve a certain general error rate.*
3. *it relates a general relation between learning performance and the size of the hypothesis space, and the number of training examples,*
4. *the more data we have, the lower the upper bound of error we can achieve, and*
5. *the smaller the hypothesis size is, the less data we need to achieve a certain general error rate.*

In this framework, we assumed that $\mathcal{C} \subseteq H$. This case is called realizable case.

Example (PAC-Learning of conjunctions)

1. Let $\mathcal{X} = \{0,1\}^n$ be the set of all n-bit vectors.
2. Let the concept class $\mathcal{C}$ consist of all conjunctions (AND of a subset of the (possibly negated) variables, such as $c(x) = x_2 \wedge x_7 \wedge x_9$.
3. The hypothesis space has $|H| = 3^n$ different hypotheses, then the sample complexity

$$\begin{aligned} m_H(\epsilon,\delta) &\geq \frac{1}{\epsilon}\left(\log|H| + \log\left(\frac{1}{\delta}\right)\right) \\ &= \frac{1}{\epsilon}\left(\log 3^n + \log\left(\frac{1}{\delta}\right)\right) \\ &= \frac{1}{\epsilon}\left(n\log 3 + \log\left(\frac{1}{\delta}\right)\right) \end{aligned}$$

4. This means that a polynomial number of samples will do to get a good enough hypothesis with high enough probability.
5. Therefore the class of conjunctions is **PAC learnable**.
6. Let $\delta = 0.02$, $\epsilon = 0.1$, and $n = 10$. Then $m_H(\epsilon,\delta) \geq 149$.
7. The computation complexity cost per training example is in $O(n)$.

Example (PAC-Learning of DNF)

1. Let $\mathcal{X} = \{0,1\}^n$ be the set of all n-bit vectors.
2. Let the concept class $\mathcal{C}$ consist of all DNF that is the OR of an arbitrary number of arbitrary-length conjunctions.
3. For example, we have $c(x) = (x_2 \wedge x_7 \wedge x_8) \vee (x_4 \wedge \bar{x}_9) \vee (x_3 \wedge \bar{x}_5 \wedge x_7)$.
4. The hypothesis space has $|H| = 2^{2^n}$ different hypotheses, then the sample complexity

$$\begin{aligned} m_H(\epsilon,\delta) &\geq \frac{1}{\epsilon}\left(\log|H| + \log\left(\frac{1}{\delta}\right)\right) \\ &= \frac{1}{\epsilon}\left(\log 2^{2^n} + \log\left(\frac{1}{\delta}\right)\right) \\ &= \frac{1}{\epsilon}\left(2^n \log 2 + \log\left(\frac{1}{\delta}\right)\right) \end{aligned}$$

5. This does not tell us that the class is PAC learnable, because the bound is not a tight bound.
6. This bound only shows that we cannot use the general learning bound to show PAC learnability.
7. **The question of whether this class is PAC learnable or not is an open problem.**

Agnostic probably approximately correct model

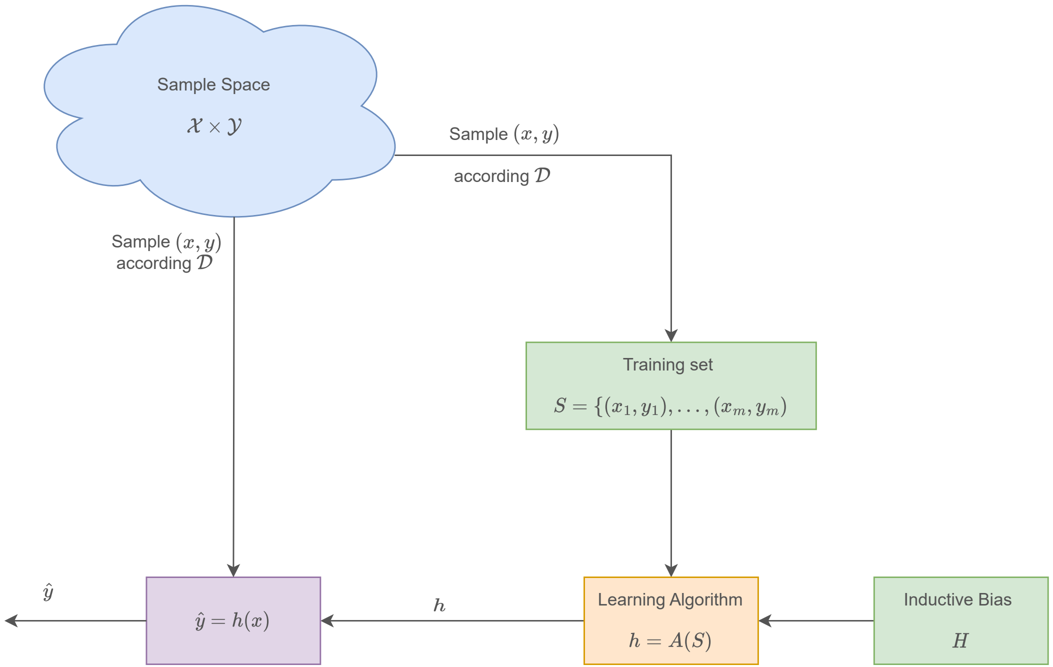
1. We assumed that there exists a target function $c \in \mathcal{C} \subseteq H$ that perfectly labels the data.
2. This is not always a valid assumption to make.
   - There could be random noise in the data, sometimes causing a label to be flipped.
   - There is a perfect target function c, but it is not in hypotheses space that we are considering, i.e. $c \notin H$.
   - There is so much randomness in the labels that no function comes close to labeling all of our data correctly. In this, we would like to remove the assumption of a perfect target function.
3. This is often referred to as the agnostic learning setting, since we make no assumptions about the origin of labels.
4. It is also referred to as unrealizable setting, in contrast with realizable setting.
5. We need to update all of our definitions, assumptions, and goals for this new setting.

1. We now assume that there exists a joint distribution $\mathcal{D}$ over pairs of values $(\mathbf{x}, y)$ where $\mathbf{x}$ is the input point and y is the corresponding label.
2. We now need to update our notion of error.

$$\mathbf{R}(h) = \underset{(\mathbf{x},y)\sim\mathcal{D}}{\mathbb{P}}[h(x) \neq y]$$

3. We can still model a perfect target function as a joint probability distribution for which the label y is deterministically equal to $c(\mathbf{x})$ conditioned on the input $\mathbf{x}$.
4. Previously, our goal was to find a hypothesis $h \in H$ such that $\mathbf{R}(h) \leq \epsilon$. Here, such a function might not exist.
5. We can only possibly hope to find a function as good as the best function $h \in H$.
6. Therefore, our new goal is to output a function $h \in H$ such that $\mathbf{R}(h)$ is close to $\min_{h' \in H} \mathbf{R}(h')$.

Agnostic PAC learning model



Definition (Agnostic PAC Learnability)

A hypothesis class H is **agnostic PAC learnable**, if there exist a function $m_H : (0,1)^2 \mapsto \mathbb{N}$ and a learning algorithm $\mathcal{A}$ with the following property: for every $\epsilon, \delta \in (0,1)$ and for every distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$, when running the learning algorithm $\mathcal{A}$ on $m \geq m_H(\epsilon, \delta)$ i.i.d. examples generated by $\mathcal{D}$, the algorithm returns a hypothesis $h \in H$ such that, with probability of at least $(1-\delta)$ the following equation holds (over the choice of the m training examples),

$$\mathbf{R}(h) \leq \min_{h' \in H} \mathbf{R}(h') + \epsilon$$

1. If the realizability assumption holds, then agnostic PAC learning provides the same guarantee as PAC learning. In that sense, agnostic PAC learning generalizes the definition of PAC learning.
2. When the realizability assumption does not hold, no learner can guarantee an arbitrarily small error. A learner can success if its error is not much larger than the best error achievable by a predictor from the class H while in PAC learning the learner is required to achieve a small error in absolute term.

Definition (Generalized loss function)

Given any set H and some domain $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$, let ℓ be any function from $H \times \mathcal{Z}$ to the set of nonnegative real numbers, $\ell : H \times \mathcal{Z} \mapsto \mathbb{R}_+$. We call such functions **loss functions**.

Definition (Risk function)

The risk function is the expected loss of $h \in H$ with respect to a probability distribution $\mathcal{D}$ over $\mathcal{Z}$,

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

Definition (Agnostic PAC Learnability for general loss functions)

A hypothesis class H is **agnostic PAC learnable** with respect to a set $\mathcal{Z}$ and a loss function $\ell : H \times \mathcal{Z} \mapsto \mathbb{R}_+$, if there exist a function $m_H : (0,1)^2 \mapsto \mathbb{N}$ and a learning algorithm $\mathcal{A}$ with the following property: for every $\epsilon, \delta \in (0,1)$ and for every distribution $\mathcal{D}$ over $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$, when running learning algorithm $\mathcal{A}$ on $m \geq m_H(\epsilon, \delta)$ i.i.d. examples generated by $\mathcal{D}$, the algorithm returns a hypothesis $h \in H$ such that, with probability of at least $(1 - \delta)$ the following equation holds,

$$\mathbf{R}(h) \leq \min_{h' \in H} \mathbf{R}(h') + \epsilon$$

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

Uniform convergence

Definition (Empirical risk minimization algorithm)

Let H be the hypotheses set and S be the training set. An empirical risk minimization algorithm receives a training set S and a hypotheses set H and outputs a hypothesis $h \in H$ such that

$$h = \arg\min_{h' \in H} \hat{\mathbf{R}}(h').$$

1. Given a hypothesis class, H, an ERM algorithm receives a training sample, S and evaluates the risk of each $h \in H$ on S and outputs a member of H that minimizes this empirical risk.
2. The hope is that an h that minimizes the empirical risk with respect to S is a risk minimizer (or has risk close to the minimum) with respect to the true data probability distribution as well.

1. **It suffices to ensure that the empirical risks of all members of H are good approximations of their true risk.**
2. In another word, we need that uniformly over all hypotheses in the hypothesis class, the empirical risk will be close to the true risk.

Definition (ϵ-representative sample)

A training set S is called ϵ-representative (w.r.t. domain $\mathcal{Z}$, hypothesis class H, loss function ℓ, and distribution $\mathcal{D}$) if $\forall h \in H$, we have

$$|\mathbf{R}(h) - \hat{\mathbf{R}}(h)| \leq \epsilon.$$

Lemma

Let the training set S is $\frac{\epsilon}{2}$-representative (w.r.t. domain $\mathcal{Z}$, hypothesis class H, loss function ℓ, and distribution $\mathcal{D}$). Then, any output of ERM algorithm (h_s), satisfies

$$\mathbf{R}(h_s) \leq \min_{h \in H} \mathbf{R}(h) + \epsilon.$$

Proof.

For every $h \in H$, we have

$$\begin{aligned}
\mathbf{R}(h_s) &\leq \hat{\mathbf{R}}(h_s) + \frac{\epsilon}{2} && \text{By applying the } \frac{\epsilon}{2}\text{-representativeness of } S \text{ to } h_s. \\
&\leq \hat{\mathbf{R}}(h) + \frac{\epsilon}{2} && \text{Because } h_s \text{ is an ERM predictor, hence } \hat{\mathbf{R}}(h_s) \leq \hat{\mathbf{R}}(h). \\
&\leq \mathbf{R}(h) + \frac{\epsilon}{2} + \frac{\epsilon}{2} && \text{Because } S \text{ is } \frac{\epsilon}{2}\text{-representative, so } \hat{\mathbf{R}}(h) \leq \mathbf{R}(h) + \frac{\epsilon}{2}. \\
&= \mathbf{R}(h) + \epsilon
\end{aligned}$$

□

This lemma implies that to ensure that the ERM rule is an agnostic PAC learner, it suffices to show that with probability of at least $(1 - \delta)$ over the random choice of a training set, it will be an ϵ-representative training set.

Definition (Uniform convergence)

A hypothesis class H has **uniform convergence property** (w.r.t a set $\mathcal{Z}$ and a loss function ℓ), if there exist a function $m_H^{UC} : (0,1)^2 \mapsto \mathbb{N}$ such that for every $\epsilon, \delta \in (0,1)$ and for every probability distribution $\mathcal{D}$ over $\mathcal{Z}$, if S is a sample of $m \geq m_H^{UC}(\epsilon, \delta)$ examples drawn i.i.d according to $\mathcal{D}$, then with probability of at least $(1-\delta)$, the training set S is ϵ-representative.

The term **uniform** here refers to having a **fixed sample size** that works for all members of H and over all possible probability distributions over the domain.

Here, we used the fact that for every $h \in H$, the empirical risk concentrates around the true risk with high probability. This concept known as uniform convergence.

Theorem

If a class H has the uniform convergence property with a function m_H^{UC}, then the class is **agnostically PAC learnable** *with the sample complexity $m_H(\epsilon, \delta) \leq m_H^{UC}(\frac{\epsilon}{2}, \delta)$.*

Proof.

1. Suppose that H has the uniform convergence property with a function m_H^{UC}.
2. For every $\epsilon, \delta \in (0, 1)$, if S is a sample of size m, where $m \geq m_H^{UC}(\frac{\epsilon}{2}, \delta)$, then with probability at least $(1 - \delta)$, sample S is $\frac{\epsilon}{2}$−representative. This means that for all $h \in H$ we have

$$\mathbf{R}(h) \leq \hat{\mathbf{R}}(h_s) + \frac{\epsilon}{2},$$

or

$$\begin{aligned} \mathbf{R}(h) &\leq \min_{h' \in H} \hat{\mathbf{R}}(h') + \frac{\epsilon}{2} \\ &\leq \min_{h' \in H} \mathbf{R}(h') + \epsilon \end{aligned}$$

3. Hence H is agnostically PAC-learnable with $m_H(\epsilon, \delta) = m_H^{UC}(\frac{\epsilon}{2}, \delta)$.

□

Agnostic PAC-Learning for finite H

Theorem

Let H be a finite hypothesis class. Then, H enjoys the uniform convergence property with sample complexity

$$m_H^{UC}(\epsilon, \delta) \leq \frac{\ln\left(\frac{2|H|}{\delta}\right)}{2\epsilon^2}$$

and is therefore PAC learnable by the ERM algorithm.

Theorem (Hoeffding inequality)

Let $\theta_1, \ldots, \theta_m$ be be a sequence of i.i.d. random variables and assume that for all i, we have $\mathbb{E}[\theta_i] = \mu$ and $\mathbb{P}[a \leq \theta_i \leq b] = 1$. Then, for any $\epsilon > 0$

$$\mathbb{P}\left[\left|\frac{1}{m}\sum_{i=1}^{m}\theta_i - \mu\right| > \epsilon\right] \leq 2\exp\left(-\frac{2m\epsilon^2}{(b-a)^2}\right)$$

Proof.

To show that uniform convergence holds we follow a two step argument.

1. The first step applies the union bound.
   - 1.1 Fix some ϵ, δ.
   - 1.2 We need to find a sample size m that guarantees that for any $\mathcal{D}$, with probability of at least $(1-\delta)$ of the choice of S sampled i.i.d. from $\mathcal{D}$, for all $h \in H$ we have $|\hat{\mathbf{R}}(h) - \mathbf{R}(h)| \leq \epsilon$.
   - 1.3 That is,

$$\mathbb{P}\left[\forall h \in H \;\middle|\; \left|\hat{\mathbf{R}}(h) - \mathbf{R}(h)\right| \leq \epsilon\right] \geq (1-\delta)$$

   - 1.4 Equivalently, we need to show that

$$\mathbb{P}\left[\exists h \in H \;\middle|\; \left|\hat{\mathbf{R}}(h) - \mathbf{R}(h)\right| > \epsilon\right] < \delta$$

$$\bigcup_{h \in H} \mathbb{P}\left[\left|\hat{\mathbf{R}}(h) - \mathbf{R}(h)\right| > \epsilon\right] < \delta$$

$$\bigcup_{h \in H} \mathbb{P}\left[\left|\hat{\mathbf{R}}(h) - \mathbf{R}(h)\right| > \epsilon\right] < \sum_{h \in H} \mathbb{P}\left[\left|\hat{\mathbf{R}}(h) - \mathbf{R}(h)\right| > \epsilon\right] < \delta$$

Proof (Cont.).

2. The second step employs a measure concentration inequality.
   2.1 This step will argue that each summand of the right-hand side of this inequality is small enough.
   2.2 That is, we will show that for any fixed hypothesis, h, value of $\left|\hat{\mathbf{R}}(h) - \mathbf{R}(h)\right|$ is likely to be small.
   2.3 Recall that

$$\mathbf{R}(h) = \mathop{\mathbb{E}}_{z \sim \mathcal{D}} [\ell(h, z)]$$

$$\hat{\mathbf{R}}(h) = \frac{1}{m} \sum_{i=1}^{m} \ell(h, z_i)$$

Because each z_i is sampled i.i.d. from $\mathcal{D}$.

   2.4 By the linearity of expectation, it follows that

$$\mathbf{R}(h) = \mathbb{E}\left[\hat{\mathbf{R}}(h)\right]$$

   2.5 Hence, quantity $\left|\hat{\mathbf{R}}(h) - \mathbf{R}(h)\right|$ is deviation of random variable $\hat{\mathbf{R}}(h)$ from its expectation.
   2.6 We must show that $\hat{\mathbf{R}}(h)$ is concentrated around its expected value.

Proof (Cont.).

3. The second step employs a measure concentration inequality (cont.).
   3.1 Let $\theta_i = \ell(h, z_i)$.
   3.2 Since h is fixed and $z_1, \ldots, z_m$ are sampled i.i.d., then $\theta_1, \ldots, \theta_m$ are also i.i.d. random variables. Hence,

$$\hat{\mathbf{R}}(h) = \frac{1}{m}\sum_{i=1}^{m}\theta_i \, \mathbb{P}\left[\left|\frac{1}{m}\sum_{i=1}^{m}\theta_i - \mu\right| > \epsilon\right]$$
$$\mathbf{R}(h) = \mu$$

   3.3 Also assume that $\ell \in [0, 1]$, then $\theta_i \in [0, 1]$.
   3.4 Using Hoeffding's inequality

$$\begin{aligned}
\mathbb{P}\left[\exists h \in H \;\middle|\; \left|\hat{\mathbf{R}}(h) - \mathbf{R}(h)\right| > \epsilon\right] &= \sum_{h \in H} \mathbb{P}\left[\left|\frac{1}{m}\sum_{i=1}^{m}\theta_i - \mu\right| > \epsilon\right] \\
&\leq \sum_{h \in H} 2\exp\left(-2m\epsilon^2\right) \\
&= 2|H|\exp\left(-2m\epsilon^2\right) \leq \delta
\end{aligned}$$

   3.5 Solving the above inequality completes the proof of the theorem.

□

Summary

Summary

1. We have shown that **finite hypothesis classes enjoy the uniform convergence property** and are hence **agnostic PAC learnable**.
2. What happen if $|H|$ is not finite?

Reading

Readings

1. Chapters 3 & 4 of Understanding machine learning : From theory to algorithms (Shalev-Shwartz and Ben-David 2014).
2. Chapter 2 of Foundations of machine learning (Mohri, Rostamizadeh, and Talwalkar 2018).

Mohri, Mehryar, Afshin Rostamizadeh, and Ameet Talwalkar (2018). *Foundations of Machine Learning*. Second Edition. MIT Press.

Shalev-Shwartz, Shai and Shai Ben-David (2014). *Understanding machine learning: From theory to algorithms*. Cambridge University Press.

Questions?