Indian Institute of Technology Kanpur
CS777 Topics in Learning Theory

Scribe: Abhibhav Garg
Instructor: Purushottam Kar
Date: January 15, 2018

LECTURE 4

Notation, Learning Algorithms and Convergence

1 Introduction

Over the past three lectures, we have motivated the problem of "learning" by considering a most simple toy example. The example was as follows: Given two coins, pick the one with higher bias. The obvious method of doing so is to toss both the coins a number of times and pick the one with the empirical bias, the only question is how many times should the coins be tossed before we are certain of our choice. To answer this, we studied and used a concentration inequality - namely the Chernoff bound. There is an obvious parallel between this example, and the problem of picking a model for our learning problem - in both cases we have to decide between (possibly infinite) candidates, by observing their performance.

Thus motivated, we now look at the learning problem in its full generality. This lecture begins by setting some notation that will be used for the rest of the course. Following this, we look at the model selection problem in a more general setting, and show, using the concept of convergence, why this is a hard problem.

2 Notation and Setting

Following is some notation that will be used throughout the course. Readers are encouraged to read Chapter 2 from Shalev-Shwartz and Ben-David (2016) for further information.

- $\mathcal{X}$ denotes the instance/feature space. Typically, $\mathcal{X} \subseteq \mathbb{R}^d$. For example, $\mathcal{X} \subseteq \mathcal{B}_1^d(0)$ for (normalized) regression problems, $\mathcal{X} \subseteq [0,1]^d$, when the inputs consist of d pixel sized grey-scale images.
- $\mathcal{Y}$ denotes the output space. Examples include $\mathcal{Y} = \{-1, 1\}$ for classification problems, $\mathcal{Y} \subseteq \mathbb{R}^p$ for general regression problems, $\mathcal{Y} = S_n$, the permutation group, for ranking problems, $\mathcal{Y} =$ the set of all parse trees, for parsing problems.
- $\mathcal{S}$ denotes the training data. It is of the form $((x_1, y_1), \ldots, (x_n, y_n)) \in (\mathcal{X} \times \mathcal{Y})^n$, for some n.

We now use this notation to fully describe a general learning problem. Here, the focus is on the supervised learning setting,

Definition 4.1 (Hypothesis Space). $\mathcal{F} \subseteq \mathcal{Y}^{\mathcal{X}}$, a subset of all possible functions of the form $f : x \to y, x \in \mathcal{X}, y \in \mathcal{Y}$ is the set of all functions we are interested in. $\mathcal{F}$ is called the *hypothesis space* or the *model space*. Examples include the set of all linear functions (when we are interested in linear regression), and the set of all functions realizable by a neural network of a particular shape and size.

Remark 4.1. The range of the functions in our hypothesis space can possibly be different from the actual output space. For example, in classification problems, the actual output(the label) is a discrete value from a set of possibilities, while the output of the learning algorithm may be a probability distribution over the labels. We thus represent it by $\hat{\mathcal{Y}}$, whenever the difference is significant.

The assumption that is made on the training data is the following: There exists a distribution $\mathcal{D}$, over $(\mathcal{X}, \mathcal{Y})$ such that $S \sim \mathcal{D}^n$. In other words, $\mathcal{S}$ consists of independent and identically distributed(i.i.d) samples from $\mathcal{D}$. This setting is called the *noisy* or *agnostic* setting. Alternatively, we can assume that the distribution $\mathcal{D}$ is only over $\mathcal{X}$, and the training data is generated as $((x_1, f^*(x_1)), (x_2, f^*(x_2)), \ldots, (x_n, f^*(x_n))), (x_1, \ldots, x_n) \sim \mathcal{D}^N, f^* \in \mathcal{Y}^{\mathcal{X}}$. This is called the *realizable* setting. We do not require f^* to belong to our hypothesis space.

Remark 4.2. Note the difference between the two settings. In the noisy setting, it is possible for the training data to consist of two copies of the same feature vector, with different labels. This is akin to asking humans to label some data - different people may give different labels to the same image.

On the other hand, in the realizable setting, if a particular feature vector appears twice, it will have the same label both times.

Definition 4.2 (Loss Function)**.** A loss function is a function of the form $l : \left(\hat{\mathcal{Y}} \times \mathcal{Y}\right) \to \mathbb{R}$. For a fixed $(x, y) \in (\mathcal{X} \times \mathcal{Y})$, the loss function can be seen as a function of a member of the hypothesis space $\mathcal{F}$ as $l(f(x), y), f \in \mathcal{F}$.

Definition 4.3 (Learning Algorithm)**.** $\mathcal{A} : \bigcup\limits_{i=0}^{\infty} \{\mathcal{X} \times \mathcal{Y}\}^i \to \mathcal{F}$ is a function from all possible training data to the hypothesis space, and is called the *learning algorithm.*

Once we have notions of loss functions and learning algorithms, we can start evaluating how functions from our hypothesis class perform on some data. The learning algorithm can then be seen as a function that takes the data as the input, and outputs a function that optimizes the loss functions. Naturally, we would like a function that not only optimizes the loss function on the given training data, but on the entire distribution $\mathcal{D}$. In order to characterize this difference, we make the following definitions:

Definition 4.4 (l-risk)**.** The *l-risk*, more commonly known as the *test error* is defined as follows:

$$e^l_{\mathcal{D}}[f] \triangleq \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}} [l(f(x), y)]$$

Definition 4.5 (Empirical risk)**.** The *empirical risk*, more commonly known as the *training error* is defined as follows:

$$e^l_{\mathcal{S}}[f] \triangleq \frac{1}{n} \sum_{i=1}^{n} l(f(x_i, y_i))$$

Claim 4.1. $\forall f \in \mathcal{F} \mathop{\mathbb{E}}_{\mathcal{S}\sim\mathcal{D}} \left[e^l_{\mathcal{S}}[f]\right] = e^l_{\mathcal{D}}[f]$

Exercise 4.1. Prove claim 4.1. *Hint:* functions preserve independence.

Example 4.1 (0-1 Loss)**.** Consider the loss function

$$l^{0-1}(\hat{y}, y) = \begin{cases} 1 & \hat{y} \neq y \\ 0 & \hat{y} = y \end{cases}$$

This is called the 0-1 loss, or the *misclassification* loss.

Exercise 4.2. Show that $e_{\mathcal{D}}^{0-1}[f] \triangleq e_{\mathcal{D}}^{l^{0-1}}[f] = \underset{(x,y)\sim\mathcal{D}}{\mathbb{P}}[f(x) \neq y]$.

In other words, show that the $0-1$ loss is equal to the probability of falsely classifying a point sampled from $\mathcal{D}$.

Example 4.2 (Generalized 0-1 Loss). We can generalize the 0-1 loss as follows:

$$l_{\alpha}^{0-1}(\hat{y}, y) = \begin{cases} \alpha & \hat{y} \neq y, y = 1 \\ 1-\alpha & \hat{y} \neq y, y = -1 \\ 0 & \hat{y} = y \end{cases}$$

In this generalization, we penalize the algorithm differently for false positives and false negatives. This can be used, for example, if we are a credit card company, and we really care about stopping fraud, and do not mind some unhappy customers.

Exercise 4.3. Find a form for $e_{\mathcal{D}}^{\alpha}[f]$ similar to that from Exercise 4.2

3 Toy Binary Classification Problem and Convergence

We now describe a toy problem, and draw parallels between said problem, and the coin game played by Alice and Bob. (See Gopakumar and Kar (2017) for details).

3.1 Problem Definition

Consider a classification problem. Let the hypothesis class be finite, $\mathcal{F} = \{f_1, \ldots, f_n\}$, where, as defined above, each f_i is a function from $\mathcal{X}$ to $\mathcal{Y} = \{-1, 1\}$. Along with this, we are also given some training data, $\mathcal{S} \sim \mathcal{D}^n$, with the promise that it is an i.i.d sample from the distribution $\mathcal{D}$.

The goal is to then try and find the "best" model from $\mathcal{F}$, that is

$$f^* = \arg\min_{f\in\mathcal{F}} e_{\mathcal{D}}^{0-1}[f]$$

This is, in more common terms, trying to find the model with the best validation error. However, it is clear that this problem is potentially ill-defined, since all we are given is $\mathcal{S}$, a set of samples from $\mathcal{D}$. All we can see is the empirical risk of each function, as opposed to the l-risk, which we want to optimize.

It is clear that in the worst case, there might be functions that have very low empirical risk, but very high l-risk, a situation we want to avoid. Here, we can finally see the connection with the Alice-Bob coin toss problem. Over there, Alice had to pick the coin with the higher bias, but she could only observe the empirical bias of each coin.

However, all is not lost. Claim 4.1 tells us that the empirical risk is equal to the l-risk, under expectation. Thus, it can be used as a proxy for the l-risk, similar to how the empirical bias was used as a proxy for the actual bias of the coin. While analyzing Alices argument, we bounded the probability that the coin with the higher bias gets a lower empirical bias than the coin with the lower bias. We now formalize this, and try and get similar results in the case of the learning problem.

3.2 ϵ-Good Sets

Define

$$\hat{f} = \arg\min_{f\in\mathcal{F}} e_{\mathcal{S}}^{0-1}[f]$$

the minimizer of the empirical risk. Without loss of generality, let $f^* = f_1$, the minimizer of the l-risk. In order for our algorithm to correctly pick $\hat{f} = f_1$, we require two things to happen simultaneously:

- We do not want f_1 to perform "poorly" on $\mathcal{S}$.
- We do not want $f_2, \dots f_n$ to perform "nicely" on $\mathcal{S}$.

If these two conditions hold, it is clear that picking the empirical risk minimizing function will be good. In order to quantify "poor" and "nice" performances, we define the following:

Definition 4.6 (ϵ-Good)**.** A set $\mathcal{S}$ is said to be ϵ-good with respect to a function f, written as $\mathcal{S} \in good_f(\epsilon)$ if the following holds:

$$\left| e^l_{\mathcal{D}}[f] - e^l_{\mathcal{S}}[f] \right| \leq \epsilon$$

More verbosely, a set is *ϵ-good* with respect to a function, if its empirical risk is within an additive factor of ϵ of the l-risk - in other words (for low enough ϵ), if the set $\mathcal{S}$ is a good representative of $\mathcal{D}$, for the function f.

Exercise 4.4. Show that if $\mathcal{S} \sim \mathcal{D}^n$, then $\forall f \in \mathcal{F}, \mathbb{P}[\mathcal{S} \in good_f(\epsilon)] \geq 1 - 2\exp\left(-\frac{n\epsilon^2}{3}\right)$.
Hint: Use Chernoff bound on the loss function, treating it as bernoulli random variable.

The above condition is called *point-wise* convergence. On first glance, it seems like this fixes all our problems - For each set, the probability that a function f performs very differently on $\mathcal{D}$ and $\mathcal{S}$ is exponentially small, and thus, the empirical risk minimizing function should work. However, on careful observation, we note that the above only bounds the probability that a given set is "bad" for some particular function f. However, we require a set that is good for all the functions f_i. This is a much stronger result.

Inspired by the previous argument, we now define the notion of uniformly good sets, as follows:

Definition 4.7 (Uniformly ϵ-Good)**.** A set $\mathcal{S}$ is called *uniformly ϵ-good*, written $\mathcal{S} \in good(\epsilon)$ if it satisfies the following property:

$$\forall f \in \mathcal{F}, \mathcal{S} \in good_f(\epsilon)$$

3.3 Solving the Problem

Armed with the definition of uniformly good sets, we can now try and find the probability that picking the empirical risk minimizing function will do us good. What we require is the following:

$$\mathbb{P}\left[e^{0-1}_{\mathcal{D}}[f_{ERM}] > e^{0-1}_{\mathcal{D}}[f^*] + \epsilon\right] < \delta$$

where, f_{ERM} is the function that minimizes empirical risk, f^* is the function that minimizes l-risk(the function we would like to pick), and δ is a confidence parameter.

The above equation basically translates to the fact that we would like f_{ERM} to be more than ϵ worse than the optimal f^* with a bounded (and low) probability δ. The following analysis gives us the condition for this to hold.

Let us assume that we have an ϵ-good set(uniformly), $\mathcal{S}$. The empirical risk minimization algorithm gives us the following guarantee, by virtue of the algorithm definition itself.

$$e^{0-1}_{\mathcal{S}}[f_{ERM}] \leq e^{0-1}_{\mathcal{S}}[f^*] \tag{1}$$

Using the fact that $\mathcal{S}$ is ϵ-good, we get

$$e_{\mathcal{D}}^{0-1}\left[f_{ERM}\right] \le e_{\mathcal{S}}^{0-1}\left[f_{ERM}\right] + \epsilon$$

Substituting Equation 1 and using the fact that $\mathcal{S}$ is ϵ-good a second time, we get

$$\begin{aligned} e_{\mathcal{D}}^{0-1}\left[f_{ERM}\right] &\le e_{\mathcal{S}}^{0-1}\left[f^*\right] + \epsilon & (2)\\ &\le e_{\mathcal{D}}^{0-1}\left[f^*\right] + 2\epsilon & (3) \end{aligned}$$

This gives us the following result:

$$\mathcal{S} \in \mathit{good}(\epsilon) \Rightarrow e_{\mathcal{D}}^{0-1}\left[f_{ERM}\right] \le e_{\mathcal{D}}^{0-1}\left[f^*\right] + 2\epsilon \tag{4}$$

However, we want to upper bound the probability of $e_{\mathcal{D}}^{0-1}\left[f_{ERM}\right]$ being **greater** than $e_{\mathcal{D}}^{0-1}\left[f^*\right]$. To this end, we consider the negation of equation 4.

Remark 4.3. A logical statement is equal to its contrapositive. This means, if we have $A \Rightarrow B$, we can infer $\neg B \Rightarrow \neg A$. Further, if we have $A \Rightarrow B$, we can infer than $\mathbb{P}\left[A\right] \le \mathbb{P}\left[B\right]$. This can be seen by noting that event B occurs every time event A occurs, due to the forward implication, but event B can also potentially occur in the absence of event A.

We now apply what we have just remarked to equation 4. Taking the contrapositive of the equation, and replacing ϵ with $\frac{\epsilon}{2}$ gives us

$$e_{\mathcal{D}}^{0-1}\left[f_{ERM}\right] > e_{\mathcal{D}}^{0-1}\left[f^*\right] + \epsilon \Rightarrow \mathcal{S} \notin \mathit{good}(\frac{\epsilon}{2}) \tag{5}$$

Applying the probability bound from the same remark gives us

$$\mathbb{P}\left[e_{\mathcal{D}}^{0-1}\left[f_{ERM}\right] > e_{\mathcal{D}}^{0-1}\left[f^*\right] + \epsilon\right] \le \mathbb{P}\left[\mathcal{S} \notin \mathit{good}(\frac{\epsilon}{2})\right] \tag{6}$$

We now use a union bound to bound the right hand side of equation 6, thus completing the analysis. We have, by definition,

$$\begin{aligned} \mathbb{P}\left[\mathcal{S} \notin \mathit{good}(\frac{\epsilon}{2})\right] &= \mathbb{P}\left[\exists f \in \mathcal{F}, \mathcal{S} \notin \mathit{good}_f(\frac{\epsilon}{2})\right] & (7)\\ &= \mathbb{P}\left[\bigcup_{f\in\mathcal{F}} \mathcal{S} \notin \mathit{good}_f(\frac{\epsilon}{2})\right] & (8)\\ &\le \sum_{f\in\mathcal{F}} \mathbb{P}\left[\mathcal{S} \notin \mathit{good}_f(\frac{\epsilon}{2})\right] & (9)\\ &\le \sum_{f\in\mathcal{F}} 2\exp\left(-\frac{n\epsilon^2}{12}\right) & (10)\\ &= 2m\exp\left(-\frac{n\epsilon^2}{12}\right) & (11) \end{aligned}$$

where m is the size of hypothesis space, $|\mathcal{F}|$. Putting it all together, we get

$$\mathbb{P}\left[e_{\mathcal{D}}^{0-1}\left[f_{ERM}\right] > e_{\mathcal{D}}^{0-1}\left[f^*\right] + \epsilon\right] \le 2m\exp\left(-\frac{n\epsilon^2}{12}\right)$$

Equivalently, with probability $1-\delta$, the l-risk of the empirical risk minimizing function is within $\sqrt{\frac{12}{n}\log\frac{2m}{\delta}}$ that of the l-risk minimizing function.

3.4 Potential Problems

There are a couple of things to note over here. In the above analysis, we used the union bound, is a very weak bound. Thus, the error we get is proportional to the size of the hypothesis space, and the above methods cannot be used to analysis problems where the hypothesis space is infinitely big. In fact, the above fails even when the hypothesis space is exponentially big, devolving into a trivial result of the form a probability is less than one. Further, we have assumed throughout that the problem of picking the empirical risk minimizing function is actually tractable. The failure of this assumption, say for example in cases where the optimization problem is NP-hard, also causes our analysis to be of no practical use.

References

Govind Gopakumar and Purushottam Kar. Statistical and Algorithmic Learning Theory, Lecture 1. Technical report, Indian Institute of Technology, Kanpur, 2017.

Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: from theory to algorithms.* Cambridge University Press, 2016.