Lecture # 8: Fundamental Theorem of Statistical ML; Introduction to Optimization

Instructor: Aditya Bhaskara   *Scribe: Sona Torosyan*

CS 5966/6966: Theory of Machine Learning

February 3^{rd}, 2022

Abstract

The lecture discusses the Fundamental Theorem of Statistical ML, some of the main implications of the theorem, as well as provides introduction to Optimization.

1 Introduction

One of our goals is to make sure that a random sample S taken from distribution D is a representative sample. Showing that a random sample is representative was discussed in the previous lecture, including the challenging case when the hypothesis class is infinite. For the latter case we introduced growth function $\tau_H(m)$ defined as the maximum number of distinct ways in which hypotheses in H classify S.

Additionally, we discussed the following theorem that bounds the error of the sample S and distribution D in terms of the growth function:

1.1 theorem. *Suppose $\tau_H(m)$ be the growth function of a hypothesis class H. Then for any X, D, if we take a sample S of size m, with probability $1 - \delta$,*

$$sup_{h \in H} |err(h, S) - err(h, D)| \leq \frac{4 + \sqrt{log \tau_H(2m)}}{\delta \sqrt{2m}}$$

This theorem can be applied for infinite hypothesis classes, and informally can be described by saying that if the growth function is small enough, then a random sample of size m is ϵ representative.

If the growth function is polynomial, $\tau_H(m) \approx m^d$ for some parameter d, then choosing $m \sim \frac{dlog(\frac{d}{\epsilon})}{\epsilon^2}$ leads to a special case when the right-hand-side of the theorem becomes $< \epsilon$, meaning that the random sample of size m is ϵ-representative w.p. 0.9.

However, for exponential cases, such as $\tau_H(m) = (1.5)^m$, it is impossible to know if the right-hand-side is small enough; hence, the above theorem cannot be used to show learnability.

1.2 definition. A hypothesis class $H : \{h : X \to \{0, 1\}\}$ is said to **shatter** a set $S \subseteq X$ if all $2^{|S|}$ possible classifications can be obtained using hypotheses $h \in H$.

1.3 definition. **VC dimension** is the size of largest set in X that can be shattered by H, i.e. $max\{m : \exists S$ of size m that is shattered$\}$.

2 Sauer-Shelah Lemma (Vapnik-Chervonenkis)

For finite hypothesis classes with VC dimension d we have the following lemma:

2.1 lemma. *Let H be a hypothesis class of finite VC dimension d. Then for every m we have:*

$$\tau_H(m) \leq \binom{m}{0} + \binom{m}{1} + ... + \binom{m}{d} \approx m^d$$

The lemma provides a better upper bound for growth function for large enough m.

Along with theorem 1.1, this lemma tells us that with finite VC dimension, the growth function has an upper bound of m^d, which itself implies that choosing $m > \frac{dlog(\frac{d}{\epsilon\delta})}{(\epsilon\delta)^2}$ gives us an ϵ representative sample. The proof of the lemma is done through inductive argument.

3 Fundamental Theorem of (Stat) Learning Theory

As a consequence of Lemma 2.1 and Theorem 1.1 we have the following theorem called *Fundamental theorem of Stat Learning Theory*:

3.1 theorem. *The following statements are equivalent:*

- *Class H is PAC learnable.*
- *Class H is agnostically PAC learnable.*
- *Class H has finite VC dimension.*

The above theorem implies that if H has infinite VC dimension, then it is not PAC learnable. This claim is proven similarly to the no-free-lunch theorem.

To prove that VC dimension is infinite, you show that for any $m \in N, \exists$ an S of size m that can be shattered.

PAC learning by default, or in the "realizable case", means that H is a given hypothesis class and if true labeling function is some $h \in H$ then we can find some h' such that $risk(h') \leq \epsilon$. In other words, we are guaranteed to find a function h' where risk is 0 if we are given $(x, h(x))$ where h is in the class H.

However, in agnostic PAC-learning case nothing can be assumed about the true labeling function. Instead, given H hypothesis class for any true label function f, we can find h' such that $risk(h') \leq min_{h \in H}(risk(h) + \epsilon)$.

PAC-learnability implies that if we could solve ERM using $\frac{log(\frac{1}{\epsilon})}{\epsilon^2}$ then error will be at most ϵ worse than best separator. However, solving ERM is harder in non-realizable case than in realizable.

The Fundamental theorem and its proof also imply that ERM is all we need assuming enough samples. Additionally, in terms of sample complexity, the agnostic case is usually as hard as the realizable case. Finally, the learnability guarantees mentioned in the theorem, only apply to ERM, not to other improper learners. In particular, performing an "improper learning" and obtaining h with 0 training-error does not imply that $h \in H$ and hence, the theorem does not apply in this case.

4 Optimization

In the next few lectures, we will look at some optimization methods. Due to the ERM problem being NP-hard, most of these methods are not guaranteed to find optima except for some settings.

Let's look at linear classification problem in general d dimension case as a start to discussing optimization. Assume, we are given $(x, f(x))$ points where $x \in R^d$ is a feature vector, and consider $H = \{h$ of the form sign $(\langle a, x\rangle + b)$ for some $a \in R^d$ and $b \in R$, $x = (x_1, x_2, ..., x_d)\}$. It can be shown that the VC-dimension of H is $d + 1$. Given true label function, VC theory says that with $\frac{d}{\epsilon^2}$ samples, we can find the "best" linear classifier for any D data distribution and any ground truth f. If $f \in H$, i.e. in the realizable case, the problem can be solved through linear programming by defining constraints in the form: $\langle a, x^{(1)}\rangle + b > 0$, $\langle a, x^{(2)}\rangle + b < 0$ assuming label of x_1 is positive and x_2 is negative.

However, in the non-realizable case the problem is NP-hard when we try to achieve an error of $\frac{1}{2} - \delta$ (i.e. any error less than half). Therefore, in ML, instead of trying to solve the binary ERM problem, we use loss functions as proxy for ERM.

In the next classes, we will look into some common loss functions and how to optimize them.