CS 596: Theoretical Machine Learning

Lecturer: Pranjal Awasthi
Scribe: Aditi Dudeja

Lecture #3
September 12, 2017

In the previous class, we had described a universal learning algorithm, called ERM (Empirical Risk Minimization). Under the ERM principle, we want to learn a function f, and we are promised that this function belongs to the function class H. We are given a training set S, and the ERM algorithm returns a function $h \in H$ that minimizes the error on the training sample. That is, $h = \text{argmin}_{g \in H} \text{err}_S(g)$. Our hope is that the function would also do well on the future data from the same distribution. We also proved the following important theorem about ERM:

Theorem 1. *Given a function class H, ERM PAC learns any $f \in H$ for any δ and for any ϵ provided that the size of the training set $m \geq \frac{1}{\epsilon} \log(\frac{|H|}{\delta})$.*

However there are some questions that this theorem does not answer:

1. How do we efficiently implement ERM?
2. The guarantee in Theorem 1 holds contingent on the promise that $f \in H$. What if $f \notin H$? Or if we don't know what H is, which is the case in most practical situations?

We will answer these questions in this lecture.

Efficiently Implementing ERM

While considering the complexity of implementing ERM, we aim to optimize some resources. These resources are as follows:

- **Sample Complexity:** This basically refers to the amount of training data. We want to be able to implement ERM using as few training points as possible.
- **Computational Complexity:** We want to be able optimize the run time of ERM.

Usually, these two resources cannot be optimized simultaneously. As an example, consider the problem of learning decision trees efficiently. As we had seen in the previous class, decision trees can be learnt in time polynomial in n and m. Here, n and m are the number of variables and the size of the training set respectively. However, by Theorem 1, we know that in order to learn decision trees, we need at least $\frac{2^n}{\epsilon} \log(\frac{1}{\delta})$ training points.

Suppose now, we know that the tree is a $\log n$ depth decision tree. Is it possible to find a $\log n$ depth tree that fits a given set of training points? It turns out that this problem is in fact NP-hard. However, this doesn't rule out the possibility of existence of a polynomial time algorithm that learns a $\log n$ depth decision tree. Therefore, we have the following open problem:

Open Problem. *Is there a polynomial time PAC algorithm for $\log n$ depth decision trees?*

Using over-parameterization, it is possible to design a quasi-polynomial time PAC algorithm for learning decision trees. We will illustrate this technique using the following problem:

Question. *Consider the class H of two-term DNFs over $x_1, \ldots, x_n$. Is it possible to efficiently PAC learn a function in this class?*

Note that this problem is NP-Hard. Further $|H| = 9^n$, this means that any function $f \in H$ can be learned provided we have at least $m \geq \frac{n}{\epsilon} \log(\frac{1}{\delta})$ training samples. We now illustrate the technique of over-parameterization, and how it can be used to obtain a faster PAC algorithm for learning a function in H. Any two term DNF can be written as a 2-CNF. Consider the following example:

Let $f = (x_1 \wedge x_2) \vee (x_3 \wedge x_7)$. Then, note that f can be rewritten as $f = (x_1 \vee x_3) \wedge (x_1 \vee x_7) \wedge (x_2 \vee x_3) \wedge (x_2 \vee x_7)$. We create $O(n^2)$ new variables corresponding to each pair of literals. We replace each clause by the appropriate variable. This problem has now become a conjunction learning problem. We had seen an **efficient** algorithm for this in the last class. We use this algorithm to solve the problem. However, now our class of problems has changed. The size of the new class, H^* is 2^{n^2}. So, by Theorem 1, the number of samples we need has increased to $\frac{n^2}{\epsilon} \log(\frac{1}{\delta})$. Although we need more samples now, using over-parameterization, we have achieved computational as well as statistical efficiency.

Learning any function

We now deal with the second question that we wanted to answer, that is what do we do in situations when we don't know if $f \in H$. Note that in this case, it is possible that no $h \in H$ fits the training set perfectly. What if we are able to say that w.h.p, all functions $h \in H$ satisfy $|\operatorname{err}_S(h) - \operatorname{err}(h)| \leq \epsilon$? If this is truly the case, then we have a reason to optimize over the training set. We will see how ERM helps us achieve this guarantee in the form of the following theorem:

Theorem 2. *Let H be any function class and S be the size of the training set. Let $|S| = m$ and $f' = \operatorname{argmin}_{g \in H} \operatorname{err}_S(g)$. Then, for any δ and for any ϵ, with probability $\geq 1 - \delta$, we have that:*

(a) $\forall h \; |\operatorname{err}_S(h) - \operatorname{err}(h)| \leq \epsilon$, *provided that* $m \geq \frac{2}{\epsilon^2} \log(\frac{2|H|}{\delta})$ *and,*

(b) $\operatorname{err}(f') \leq \operatorname{err}(f_{best}) + 2\epsilon$, *where* $f_{best} = \operatorname{argmin}_{g \in H} \operatorname{err}(g)$.

To prove this theorem, we need a concentration bound. We use Hoeffding's Inequality that we state as follows:

Hoeffding's Inequality

Let $X_1, X_2, \cdots, X_n$ be independent random variables over $[0, 1]$. Let $S = \frac{1}{n} \sum_{i=1}^{n} X_i$, then

$$\Pr[|S - E(S)| \geq \epsilon] \leq 2e^{-2n\epsilon^2} \tag{1}$$

We will now use Hoeffding's Inequality to prove the first part of Theorem 2:

Proof. (Theorem 2(a)) Note that $\text{err}_S(h)$ is a random variable because the points S that we choose are random. Now in order to prove our main theorem, we want to bound the probability that some function $h \in H$ is a 'bad' function. A 'bad' function $h \in H$ is a function such that $|\text{err}_S(h) - \text{err}(h)| > \epsilon$. We want to prove that $\Pr[\exists h \in H : |\text{err}_S(h) - \text{err}(h)| > \epsilon] \leq \delta$. Let us first bound $\Pr[|\text{err}_S(h) - \text{err}(h)| > \epsilon]$ for a fixed function $h \in H$.

$$\Pr[|\underset{S}{\text{err}}(h) - \text{err}(h)| > \epsilon] = \Pr[|\frac{1}{m}\sum_{i=1}^{m} Z_i - \text{err}(h)| > \epsilon] \tag{2}$$

Here, $m = |S|$ and Z_i is an indicator random variable that takes value 1 if $h(x_i) \neq y_i$ and 0 otherwise. Note that $E(Z_i) = \text{err}(h)$. It follows that $E(\frac{1}{m}\sum_{i=1}^{m} Z_i) = \text{err}(h)$. Now we can apply Hoeffding's Inequality and we have the following bound:

$$\Pr[|\frac{1}{m}\sum_{i=1}^{m} Z_i - \text{err}(h)| > \epsilon] \leq 2e^{-2m\epsilon^2} \tag{3}$$

Applying the union bound, we have

$$\Pr[\exists h \in H : |\underset{S}{\text{err}}(h) - \text{err}(h)| > \epsilon] \leq |H|(2e^{-2m\epsilon^2}) \leq \delta \tag{4}$$

It follows that

$$m \geq \frac{1}{2\epsilon^2}\log(\frac{2|H|}{\delta}) \tag{5}$$

□

Note that this is worse than our previous bound because $\frac{1}{\epsilon}$ has become $\frac{1}{\epsilon^2}$, but our conclusion is stronger. In some sense, $\frac{1}{\epsilon}$ is the additional cost we have to pay because we don't know H.

We now prove the second part of the theorem as follows:

Proof. (Theorem 2(b)) Let $f' = \text{argmin}_{h \in H}\, \text{err}_S(h)$. From the first part, we know that:

$$\text{err}(f') \leq \underset{S}{\text{err}}(f') + \epsilon \tag{6}$$

$$\leq \underset{S}{\text{err}}(f_{best}) + \epsilon \tag{7}$$

$$\leq \text{err}(f_{best}) + 2\epsilon \tag{8}$$

The second inequality follows from the definition of f', and the third inequality follows from applying 2(a) to f_{best}. With this, we have proved our claim. □

Note that from the above theorem, we have the following corollary:

Corollary 1. *Given a finite function class H, for any $\epsilon > 0$, $\delta > 0$, any distribution D over $\mathcal{X}$ and any target function h^*, with probability $\geq 1 - \delta$, we have that:*

$$\forall h \in H \; |\underset{S}{\text{err}}(h) - \text{err}(h)| \leq \sqrt{\frac{1}{2m}\log(\frac{2|H|}{\delta})}. \tag{9}$$

where m is the size of the training set S.

Proof. (Corollary 1) We apply Hoeffding's inequality with $\epsilon = \sqrt{\frac{1}{2m}\log(\frac{2|H|}{\delta})}$ to bound $\Pr[\exists h \in H : |\text{err}_S(h) - \text{err}(h)| > \epsilon]$. This, combined with the fact that $\Pr[\forall h \in H \, |\text{err}_S(h) - \text{err}(h)| \leq \epsilon] = 1 - \Pr[\exists h \in H : |\text{err}_S(h) - \text{err}(h)| > \epsilon]$ gives us our bound. □

1 Infinite Function Classes

Note that the guarantee in Theorem 2 applies only when H is infinite. A crucial step in the proof is the union bound, and typically we can't apply union bound if the function class is infinite. Does this mean that infinite function classes cannot be learnt? Recall that in last class we discussed an efficient PAC learning algorithm for the class of threshold functions. This is an infinite class. This tells us that maybe size of a function class isn't a great measure of it's complexity. In next class, we'll discuss something call the VC-dimension of a function class, a measure that accurately reflects its complexity. We will also prove the following theorem:

Theorem 3. *Given any function class H, for any $\epsilon > 0$, $\delta > 0$, for any distribution D over $\mathcal{X}$ and any target function $h^* \in H$, with probability at least $1 - \delta$, $\forall h \in H$*

$$|\operatorname*{err}_S(h) - \operatorname{err}(h)| \leq \sqrt{\frac{d}{m} \log(\frac{1}{\delta})} \tag{10}$$

where d is the VC-dimension of H.