COS 511: Theoretical Machine Learning

Lecturer: Rob Schapire
Scribe: Alexandra Palocz

Lecture #3
February 11, 2019

1 Probably Approximately Correct

Last lecture, we started looking at the Probably Approximately Correct (PAC) learning model. This is the first model we are looking at that captures what it means for an algorithm to learn, that is, to generalize to new data. This model

- Takes as input training examples with corresponding labels. These examples are independent and identically distributed (iid) over a distribution D. We want "distribution-free" learning, which means that it should work for any arbitrary D.
- The output is a hypothesis $h \in H$ that takes in some $x \sim D$ and outputs a label for it
  - Unlike with the consistency model, our hypothesis space H is not necessarily the same as the concept space C.
  - We measure the hypothesis by its generalization error, that is, its probability of misclassifying a new example:
    - $err_D(h) = Pr_{x \sim D}[(c(x) \neq h(x)]$

We consider a concept class C as being PAC-learnable by H if there exists a learning algorithm A such that, for all $c \in C$, for any distribution D, and for any error bounds $\epsilon > 0, \delta > 0$, A takes a set S of m random examples, where m is polynomial with respect to $\frac{1}{\epsilon}$ and $\frac{1}{\delta}$, and produces a hypothesis $h \in H$ such that

$$Pr[err_D(h) \leq \epsilon] \geq 1 - \delta$$

Because of the possibility of bad input, we don't just look at the probability that $err_D(h) \leq \epsilon$, but rather the probability, over the randomness of choosing S and any randomness contained in A, that we end up with a hypothesis that is approximately correct. The size of m depends on ϵ and δ since we need more data to have a better probability of more accuracy. Eventually, we will want m to be polynomial in other things as well, but for now we focus on ϵ and δ.

It is not necessarily obvious that this kind of learning is possible — after all, for something to hold to this model we would need to be able to produce a hypothesis that generalizes to a large, possibly infinite space with a relatively small (polynomial) number of examples. To this end, we show that this learning is possible with some examples.

2 Example: half-lines

We consider the world of a one dimensional real number line, where the instance space is the real numbers. Our target concept class is comprised of all positive half-lines. By this we mean that we can represent a target concept c as a real number such that any x to the right of c is labeled positive and x to the left, negative.

Now we propose a simple algorithm for learning this target class: find the minimum positive example and the maximum negative example, and then choose some point in between them to be our hypothesis h. We predict everything to the right of h as positive and everything to the left as negative. For now, we do not care how this point is chosen as long as it is within that range.

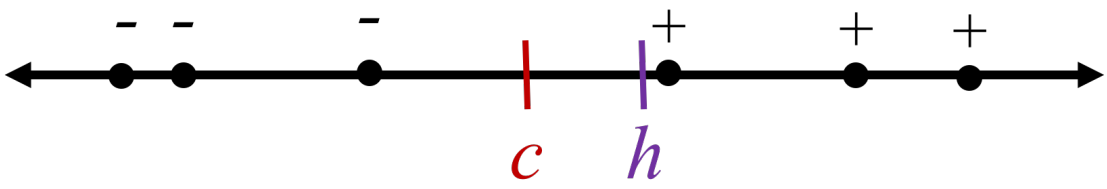

Figure 1: A positive half-line c, set of example points, and consistent hypothesis h.

We can see intuitively that this algorithm will incorrectly classify any test examples between c and h and correctly classify anything else. We want to show that this error is sufficiently small that this algorithm fits the PAC learning model — that is, that $Pr[err_D(h) \leq \epsilon] \leq 1 - \delta$.

When $err_D(h) \leq \epsilon$ does not hold, we say that h is ϵ-bad. In this case, there are two possible events where h is ϵ-bad:

- b_+, where h is at least ϵ to the right of c
- b_-, where h is at least ϵ to the left of c

In this case, we mean at least ϵ in terms of probability mass of the distribution D, not in terms of length. Let us focus on the case of b_+:

Say that we start at c and sweep to the right until we cover exactly ϵ of the probability mass (for now, we assume that D is a nice distribution where this can be done). We call this point r_+, and the region between c and r_+ we will call R_+. If b_+ is the case, then we know h must be to the right of r_+.

Claim: The b_+ event can only happen if none of the examples fall within R_+

Proof: If an example does fall within R_+, then it would be labeled as positive since it is to the right of c. However, based on the definition of A, we know that h must be to the left of all positive examples. Thus, we cannot have a case where h is to the right of r_+ and an example falls within R_+. It follows that if b_+ holds then there are no examples in R_+.

From our definition of R_+, we have $Pr[x_i \in R_+] = \epsilon$ for any one example. We can invert this to get $Pr[x_i \notin R_+] = 1 - \epsilon$.

The probability that none of the examples fall within R_+ is:

$$Pr[\text{no examples in } R_+] = Pr[(x_1 \notin R_+) \wedge \cdots \wedge (x_m \notin R_+)]$$

We use the property that $(a \Rightarrow b)$ implies that $(Pr[a] \leq Pr[b])$ along with the above to state:

$$\begin{aligned} Pr[b_+] &\leq Pr[(x_1 \notin R_+) \wedge \cdots \wedge (x_m \notin R_+)] \\ &= (1 - \epsilon)^m \leq e^{-\epsilon m} \end{aligned}$$

By symmetry, we argue that we get the same probability for the event of b_-. We can then say:

$$Pr[err_D(h) > \epsilon] \leq Pr[b_+ \vee b_-]$$

and use union bound to get

$$Pr[b_+ \vee b_-] \leq Pr[b_+] + Pr[b_-] \leq 2e^{-\epsilon m}$$

We want to know that this probability is less than our error bound δ:

$$2e^{-\epsilon m} \leq \delta$$

We can reformulate this by solving for m to get:

$$m \geq \frac{1}{\epsilon} \ln\left(\frac{2}{\delta}\right)$$

This shows that the PAC-learning criterion is satisfied if this inequality holds, that is, if the number of training examples m is at least this quantity (which is polynomial in $1/\epsilon$ and $1/\delta$).

In fact, this is even a better bound, since m is bounded by $\ln(\frac{1}{\delta})$ rather than a polynomial of $\frac{1}{\delta}$. We can also reformulate this final expression to solve for ϵ and see how the error correlates with the amount of data. With high probability, we will get:

$$err_D(h) \leq \epsilon = \frac{1}{m} \ln\left(\frac{2}{\delta}\right)$$

3 Example: Intervals

Let us consider an expansion of the previous example. We stick with the real number line, but rather than half-line, our concept class will be intervals: numbers within the interval are labeled as positive and numbers outside as negative.

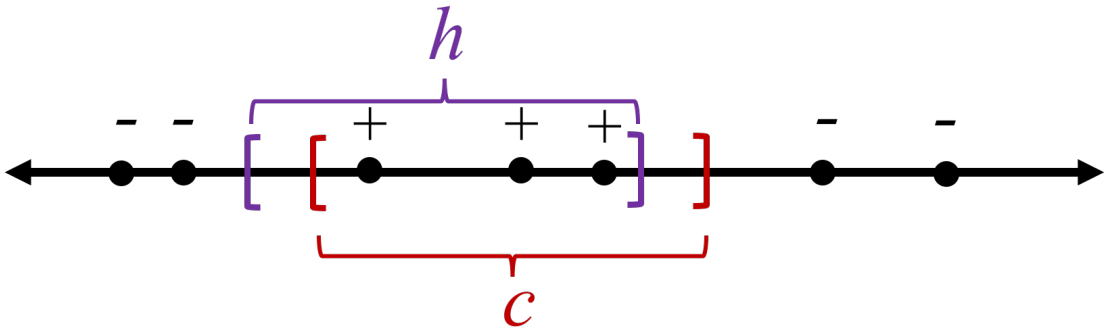

Figure 2: An interval c, set of example points, and consistent hypothesis h.

We can imagine here a similar argument to the one above, where we want h to be within $\epsilon/2$ on both sides, making the total error at most ϵ. There would then be four 'bad' events, one for each direction the hypothesis could be off on both sides of the interval. We could then go though a similar derivation as above to achieve a similar result.

4 Example: Axis-aligned rectangles

We consider a third example, recalling the concept class of axis-aligned rectangles in the plane $\mathbb{R}^2$. The example space is comprised of points; if those points are contained within the rectangles, they are labeled positive, otherwise they are labeled negative.

It is easier here to make the argument with a concrete algorithm in mind, rather than the approach above, where we were fine with any algorithm that chose an h within the minimum positive and maximum negative points. We will use the same algorithm we considered last class for a similar problem; we choose the smallest possible axis-aligned rectangle that contains all the positive points as our hypothesis.

Once again, we can imagine this as an expansion of the half-lines. We can imagine the space between our hypothesis rectangle and the concept rectangle as our space of error. We consider each dimention separately, splitting this space into four overlapping rectangles, and use a similar argument to the one above with $\epsilon/4$ instead of $\epsilon/2$. We will not go into the full argument here, but hopefully this gives some intuition as to how the results above could generalize to more dimentions.

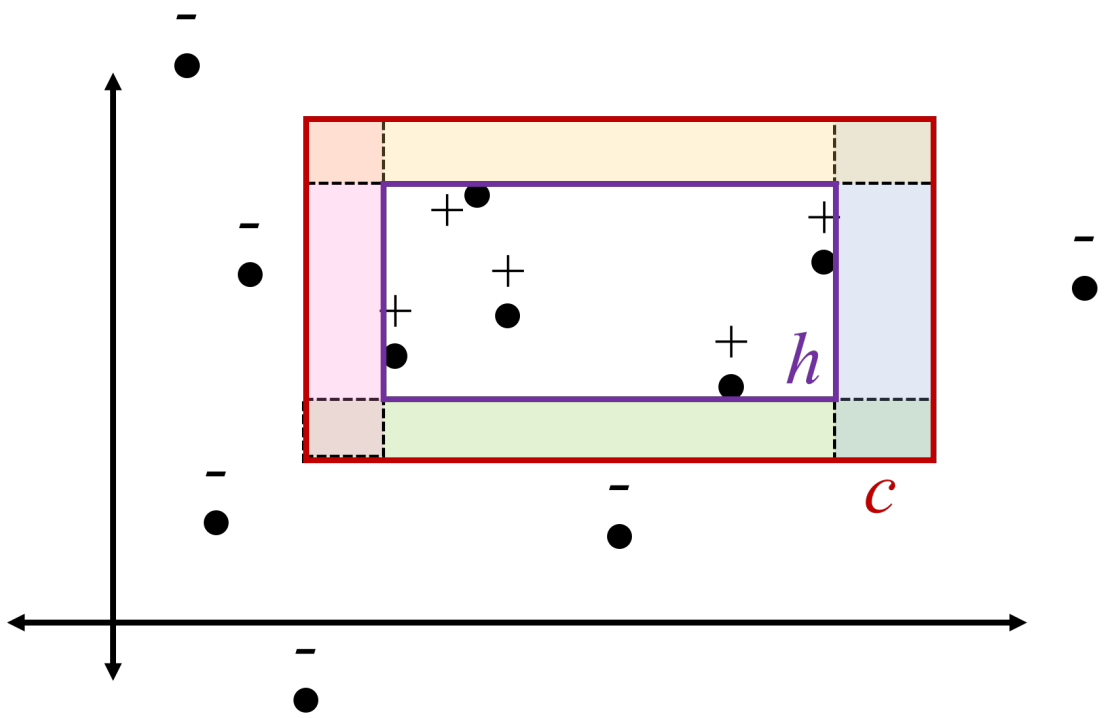

Figure 3: The error space between axis-aligned rectangle c and hypothesis h, split into four overlapping rectangles, one for each direction of error in each dimension.

5 Looking forward

Each of these examples are very specific; we want to work towards something much more general. We also haven't found yet how valuing consistency with examples relates to learn-

ing. We want to show that algorithms that find consistency are Probably Approximately Correct. So, next time, we will prove the following theorem:

Assume $|H| \leq \infty$, that is, that we have a finite hypothesis space. Say an algorithm A finds hypothesis $h_A \in H$ consistent with m random examples. Then we can say

$$Pr[err_D(h) > \epsilon] \geq \delta$$

if

$$m \geq \frac{1}{\epsilon}\left(\ln(|H|) + \ln\left(\frac{1}{\delta}\right)\right)$$

This bound based on the cardinality of H gives us our first measure of the complexity of H. The $\ln(|H|)$ makes intuitive sense if we think about implementing this as a program. This is because we would need $\lg(|H|)$ bits to assign a unique bit string to every h in H.

As an example of this, let us return to the case of monotone conjunctions from lecture 2. Let us say that we have an algorithm that finds a consistent monotone conjunction for any set of examples. How much data do we need to ensure that a consistent monotone conjunction h_A will have error at most ϵ?

There are 2^n possible monotone conjunctions on n variables. We can use the theorem above to say that, to ensure a high probability that the error is less than ϵ, that is, that $Pr[err_D(h) > \epsilon] \leq \delta$, we will need $m \geq \frac{1}{\epsilon}(n\ln(2) + \ln(\frac{1}{\delta}))$ examples.

As we see from this example, this theorem gives us a bound on the number of training examples needed for learning. We can also turn this around, however, and solve for ϵ: With probability $\geq 1 - \delta$, if $h_A \in H$ is consistent, then $err(h_A) \leq \frac{1}{m}(\ln(|H|) + \ln(\frac{1}{\delta}))$

These bounds tell us how complexity impacts how much data you need, and also how prior knowledge is helpful, since it lets us narrow down the set of hypotheses.