Learning Theory: Lecture Notes

Lecturer: Kamalika Chaudhuri Scribe: Qiushi Wang

October 9, 2015

1 The PAC Model

We will now introduce our first learning model – the Probably Approximately Correct (PAC) model. Before we introduce this model, let us first consider two examples of typical supervised learning problems.

Example 1: Spam Classification. Suppose we are trying to build a predictor that classifies if an email is spam or not. We have at our disposal a set of available emails, along with *labels* which indicate if the email is a spam or not. We will base our predictions on the *features* of each email, which are, for now, the list of words present in the email.

Example 2: Flu Diagnostic Tool. Suppose we are trying to build a tool to determine if a patient has flu or not, based on his or her symptoms. Such a tool could help doctors diagnose patients better. To help build a predictor, we have at our disposal a set of existing physician notes about past patients. For each patient, we have a set of *features* which are their symptoms (sore throat or not, fever or not, etc), and a *label* which indicates if this patient has a flu or not. Our goal is to build a predictor to classify future patients based on this existing data.

Both problems are classification problems. Before writing down a formal learning model, we first introduce some terms which can be used to describe aspects of such problems.

Domain Set. The domain set is the set of all objects that we would like to classify. For example, for spam classification, the domain set is the set of all emails, and for flu diagnosis, the domain set is the set of all patients. The domain set is usually denoted by $\mathcal{X}$.

Label Set. The label set is the set of all labels. For example, for spam classification we have two labels *spam* and *not spam*; for flu diagnosis, we have two labels *flu* and *not flu*. The label set is usually denoted by $\mathcal{Y}$. For this class, we will be mostly concerned with binary classification problems, where we have only two labels. The two labels will be usually denoted by 0 and 1 or $+1$ and -1.

Training Data. The learning algorithm takes as input a set of labelled examples and builds a classifier based on these examples. The input examples are called the training set or training data. The training set is usually denoted by $S = \{(x_1, y_1), \ldots, (x_m, y_m)\}$ where a tuple (x_i, y_i) denotes a labelled example (x_i is object i and y_i is its label).

Data Distribution. We usually assume that the training data (and the test data) is drawn from an underlying distribution D over $\mathcal{X} \times \mathcal{Y}$. This distribution is called the data distribution. Usually this distribution D is unknown, and we only have access to samples from it in the form of the training data and the test data.

The quality of a classifier is measured by its true error with respect to D, denoted by $\mathsf{err}_D(h)$, which is defined as follows:

$$\underset{D}{\mathsf{err}}(h) = \Pr_{(x,y)\sim D}[h(x) \neq y]$$

Observe that we cannot exactly determine what $\mathsf{err}_D(h)$ is because we never exactly know D. The error of h on a data set S is defined as follows:

$$\mathsf{err}(h, S) = \frac{1}{|S|}\sum_{i=1}^{|S|} 1(h(x_i) \neq y_i)$$

If the members of S are drawn iid from D, then, as $|S| \to \infty$, $\mathsf{err}(h, S) \to \mathsf{err}_D(h)$.

Learner's Output. The learner outputs a classifier or a prediction rule usually denoted by h. Note that h is a function from $\mathcal{X}$ to $\mathcal{Y}$.

Hypothesis Class. Usually we work with classifiers which belong to a fixed class; this class is called the hypothesis class. Examples are the hypothesis class of linear classifiers, the hypothesis class of decision trees, etc.

Separable. A distribution D over $\mathcal{X} \times \mathcal{Y}$ is said to be separable with respect to a hypothesis class $\mathcal{H}$ if there exists a h^* in $\mathcal{H}$ such that $\mathsf{err}_D(h^*) = 0$.

We are now ready to define the PAC model.

Definition 1 (Probably Approximately Correct (PAC) Model) *A hypothesis class $\mathcal{H}$ is said to be PAC-Learnable if there is an algorithm A with the following property. For all ϵ, δ, $0 \leq \epsilon, \delta \leq \frac{1}{2}$, all separable distributions D over $\mathcal{X} \times \mathcal{Y}$, if A is given ϵ, δ and $m_{\mathcal{H}}(\epsilon, \delta)$ examples from D, then with probability $\geq 1 - \delta$, it outputs a $h \in H$ with $\mathsf{err}_D(h) \leq \epsilon$.*

The parameter ϵ is necessary because we can never hope to find a perfect hypothesis based on only a finit eamount of data. The parameter δ is necessary because samples drawn from the data distribution D may be atypical with some small probability, and thus not very helpful in determining a good hypothesis.

A similar definition applies for non-separable D.

Definition 2 (Agnostic PAC Model) *A hypothesis class $\mathcal{H}$ is said to be Agnostic PAC-Learnable if there is an algorithm A with the following property. For all ϵ, δ, $0 \leq \epsilon, \delta \leq \frac{1}{2}$, all distributions D over $\mathcal{X} \times \mathcal{Y}$, if A is given ϵ, δ and $m_{\mathcal{H}}(\epsilon, \delta)$ examples from D, then with probability $\geq 1 - \delta$, it outputs a $h \in H$ with:*

$$\underset{D}{\mathsf{err}}(h) \leq \epsilon + \inf_{h^* \in \mathcal{H}} \underset{D}{\mathsf{err}}(h^*)$$

Example 1: Rectangles in the Plane. Let $\mathcal{X} = \mathbb{R}^2$, $\mathcal{H} = \{\text{axis-aligned rectangles in the plane}\}$. The data are points on the plane with labels $\{-, +\}$ and a hypothesis is a rectangle which labels anything inside as $+$ and anything outside as $-$. A simple algorithm for learning this class is be as follows.

algorithm $A(\varepsilon, \delta)$

draw $m_{\mathcal{H}}(\varepsilon, \delta)$ samples from D
output the smallest rectangle enclosing all $+$ points

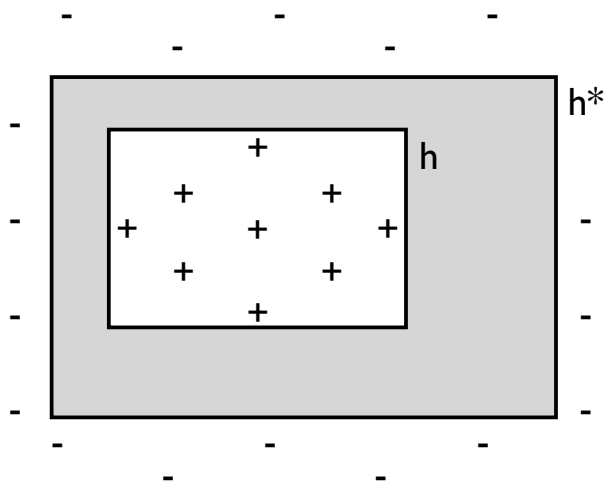

Figure 1: The true hypothesis h^* and a hypothesis h picked by the algorithm. $\mathsf{err}_D(h)$ is the probability mass in the grey shaded region.

Suppose D is separable with respect to $\mathcal{H}$. Then by assumption there must exist a rectangle h^* which perfectly seperates the $+/-$ labeled points. By construction, the h produced will always be inside h^* and A fails if the probability mass of the region enclosed by h is less than h^* by more than ε. See Figure **??**.

For any given side of h^*, look at a rectangular strip of probability mass $\varepsilon/4$ on the interior of the side. For example, the strip on the top is shaded in Figure **??**; the other strips (on the bottom, left and right) are symmetric. If each of these strips contain at least one point, then Algorithm A will not fail. For a fixed strip, this occurs with probability

$$\mathbb{P}\left(\text{strip is empty after } m \text{ samples}\right) = \left(1 - \frac{\varepsilon}{4}\right)^m \leq e^{-m\varepsilon/4}.$$

Taking a union bound over all four strips and bounding the probability of failure by δ gives $4e^{-m\varepsilon/4} \leq \delta$ meaning A needs $m \geq (4/\varepsilon)\ln(4/\delta)$ samples to succeed with probability $\geq 1 - \delta$.

Example 2: Monotone Conjunctions. Let $\mathcal{X} = \{0,1\}^n$ with $\mathcal{H} = \{\text{conjunctions of positive literals}\}$. In this case, an example $\tilde{x}$ is an assignment of n truth values to n variables $x^1, \ldots, x^n$ along with a binary label $\{0,1\}$. A hypothesis h is a subset of the literals; for an example $\tilde{x}$, $h(\tilde{x}) = 1$ if $\tilde{x}^i = 1$ for all literals x^i in h, and is 0 otherwise. Consider the following algorithm.

algorithm $A(\varepsilon, \delta)$
$h \leftarrow x^1 \wedge x^2 \wedge \cdots \wedge x^n$
repeat $m_{\mathcal{H}}(\varepsilon, \delta)$ times:
    draw a sample $(\tilde{x}, \tilde{y})$ from D
    if $\tilde{y} = 1$ remove all x^i from h for which $\tilde{x}^i = 0$

If D is separable with respect $\mathcal{H}$, there exists a hypothesis h^* with true error 0. Observe the following.

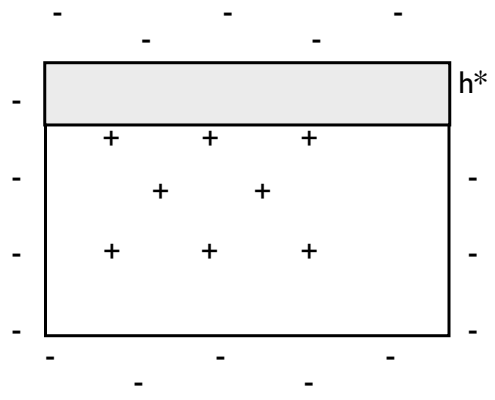

Figure 2: The true hypothesis h^* and a strip of probability mass $\varepsilon/4$ at the top inside it.

- At any time t, h contains a superset of h^*.
- If $h^*(x) = 0$ then $h(x) = 0$.
- The final h is consistent with all the seen samples (that is, it labels all the data seen so far correctly).

The output hypothesis h makes an error when it has extra variables x^i which are not in h^*. By construction of h, if x^i is an extra variable then there were no samples seen with $\tilde{x}^i = 0$ and $\tilde{y} = 1$. For a fixed i, let $q_i = \mathbb{P}_{(\tilde{x},\tilde{y})\sim D}(\tilde{x}^i = 0, \tilde{y} = 1)$ and let $I = \{i : x^i \text{ is an extra variable in } h\}$. If $q_i \leq \varepsilon/n$ for all $i \in I$ then the error is bounded by ε:

$$\operatorname*{err}_D(h) = \sum_{i \in I} \mathbb{P}_{(\tilde{x},\tilde{y})\sim D}(\tilde{x}^i = 0, \tilde{y} = 1) \leq \frac{\varepsilon}{n} \cdot |I| \leq \varepsilon.$$

On the other hand if q_i is high,

$$\mathbb{P}\left(q_i > \frac{\varepsilon}{n} \text{ but } x^i \text{ remains in } h\right) \leq \left(1 - \frac{\varepsilon}{n}\right)^m \leq e^{-m\varepsilon/n}.$$

A fails if there exists x^i not in h^* such that $q_i > \varepsilon/n$ but x^i still remains in h. By a union bound over all i, the chance of failure is at most $ne^{-m\varepsilon/n} \leq \delta$ so the algorithm will need to draw $m \geq (n/\varepsilon)\ln(n/\delta)$ samples.

Finally, if the hypothesis class is extended to allow negations of literals e.g. $\neg x^i$, the problem can be reduced to monotone conjunctions using twice as many literals in the obvious way. The same approach shows that the problem requires $m \geq (2n/\varepsilon)\ln(2n/\delta)$ samples.

1.1 A General Bound for Finite Hypothesis Classes

Theorem 1 *For all $\varepsilon > 0$, $0 < \delta < 1/2$, and any finite hypothesis class $\mathcal{H}$, if*

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

samples are drawn from a distribution D, then with probability at least $1 - \delta$, any $h \in \mathcal{H}$ consistent with the examples above has $\operatorname{err}_D(h) \leq \varepsilon$.

PROOF: Let $H = \{h \in \mathcal{H} : \mathsf{err}_D(h) > \varepsilon\}$ denote the set of bad hypotheses. The probability that a fixed $h \in H$ is consistent with a random example drawn from D is at most $1 - \varepsilon$ so

$$\begin{aligned}&\mathbb{P}\left(\text{exists } h \in \mathcal{H} \text{ such that } \underset{D}{\mathsf{err}}(h) > \varepsilon \text{ but } h \text{ is consistent with } m \text{ examples}\right)\\ &\quad\leq \sum_{h\in H} \mathbb{P}\left(h \text{ is consistent with } m \text{ examples}\right) \leq |\mathcal{H}|(1-\varepsilon)^m \leq |\mathcal{H}|e^{-m\varepsilon} \leq \delta.\end{aligned}$$

Solving for m yields the desired result. □

Applying this to the previous examples yield improved bounds on the samples needed. In the case of monotone conjunctions, $|\mathcal{H}| = 2^n$ giving a bound of $m \geq (n/\varepsilon)\ln(2/\delta)$. For conjunctions with negations, $|\mathcal{H}| = 3^n$ yielding a bound of $m \geq (n/\varepsilon)\ln(3/\delta)$.

Example 3: Decision Lists. A decision list consists of a set of literals and a sequence of binary checks:

if ℓ_1 then output b_1
else if ℓ_2 then output b_2
...
else if ℓ_n then output b_n
else output b_{n+1}

Each ℓ_i checks a single literal or its negation and either outputs $b_i \in \mathcal{Y}$ or moves to the next step. It can be assumed without loss of generality that each literal is checked exactly once. Let $\mathcal{X} = \{0,1\}^n$ be an assignment on the literals with $\mathcal{H}$ being the set of possible decision lists. This hypothesis class is finite with size $|\mathcal{H}| = n!4^n$. Consider the following algorithm.

algorithm $A(\varepsilon, \delta)$
draw $O((n\log n + \log(1/\delta))/\varepsilon)$ samples S from D
start with an empty decision list
while S is nonempty
    find any rule consistent with S that applies to at least one example
    add the rule to the bottom of the list
    remove the examples classified by this rule

This algorithm produces a list consistent with all the examples so is guaranteed to work by the general bound for finite hypothesis classes.

1.2 Occam's Razor

Another way to write the bound in Theorem **??** is:

$$\varepsilon(m) \leq \frac{\ln(|\mathcal{H}|/\delta)}{m} \tag{1}$$

Equation **??** implies that for finite hypothesis classes, larger classes $\mathcal{H}$ are more difficult to generalize than smaller ones. This phenomenon is called Occam's Razor.

Theorem 2 *Suppose we have a sequence of hypothesis classes $\mathcal{H}_1, \mathcal{H}_2, \ldots, \mathcal{H}_K$ of growing size. Suppose an algorithm draws m examples from a data distribution D and returns $h \in \mathcal{H}_{k^*}$ consistent with the samples. Then with with probability at least $1-\delta$, $\mathsf{err}_D(h) \leq (1/m)\ln(|\mathcal{H}_{k^*}|k^*(k^*+1)/\delta)$.*

PROOF: Define $\varepsilon_k = (1/m)\ln(|\mathcal{H}_k|k(k+1)/\delta)$.

$$\mathbb{P}\left(\text{exists } k,\ h \in \mathcal{H}_k \text{ such that } \operatorname*{\mathsf{err}}_D(h) > \varepsilon_k \text{ but } h \text{ is consistent with } m \text{ samples}\right)$$
$$\leq \sum_k \sum_{h \in \mathcal{H}_k} (1-\varepsilon_k)^m \leq \sum_k |\mathcal{H}_k| e^{-m\varepsilon_k} \leq \sum_k \frac{\delta}{k(k+1)} \leq \delta.$$

Here the first step follows from an union bound, and the rest from simple algebra. □

Example 4: Sparse Conjunctions. Let $\mathcal{H}_k = \{\text{class of conjunctions of } k \text{ variables out of } n\}$ and $\mathcal{H} = \bigcup_k \mathcal{H}_k$. In this case, $|\mathcal{H}_k| = \binom{n}{k} \leq n^k$. Thus, $\varepsilon_k = \frac{1}{m}(k \log n + k \log k(k+1) + \log(1/\delta))$, which is better than the standard bound of $\varepsilon = \frac{1}{m}(n + \log(1/\delta))$ when k is small.