The PAC Learning Model

These notes are slightly edited from scribe notes in previous years.

1 PAC Learning

In the previous lecture we discussed 2 learning games, and in each case were able to get an analysis such that “with high probability, our algorithm produces a hypothesis that has small error”. We abstract the details of these games (and some more reasonable properties) in the PAC model below.

**Definition:** Algorithm $A$ PAC learns concept class $C = \cup_n C_n$:

if there exists a polynomial $p(\ldots)$ such that

$\forall n$

$\forall c \in C_n$,

$\forall$ distributions $D$ over $X_n$,

$\forall 0 < \epsilon < 1$ and $\forall 0 < \delta < 1$

If learning algorithm $A$ is given $\epsilon, \delta$ as input, and access to random training examples sampled from $D$ and labeled by $c$ (the true concept to be learned), then with probability at least $1 - \delta$, $A$ outputs a hypothesis $h \in C$ such that:

$error(h) = \Pr_D[h(x) \neq c(x)] < \epsilon$

Moreover, $A$’s running time and sample size are bounded by $p(n, \frac{1}{\epsilon}, \frac{1}{\delta}, \text{size}(c))$ (i.e. they are polynomial in those parameters.)

Note that we abstract the allowed error rate $\epsilon$, and the failure probability $\delta$. We also added two new parameters: a complexity parameter for the domain (typically number of features) and the size of the learned concept. All these can be used to quantify the complexity of the learning algorithm.

2 The Tightest Fit Learning Algorithm on Interval $[a,b]$

**Theorem:** The tightest fit learning algorithm PAC learns intervals when using $m \geq \frac{2}{\epsilon} \ln \frac{2}{\delta}$ examples.

**Proof:** The true concept interval we are trying to learn is $[a,b]$. We define two points $a+$ and $b-$ as follows:

$a+ = \arg\min_x (Pr[a, x] \geq \epsilon/2)$

$b- = \arg\max_y (Pr[y, b] \geq \epsilon/2)$
Note: both points are defined relative to distribution, not relative to the learner.

Due to our definition of a+ and b-, we have:

\[
Pr(\text{one example does not hit } [a,a+]) = 1 - \frac{\epsilon}{2}
\]

\[
Pr(\text{all } m \text{ I.I.D. examples do not hit } [a,a+]) = (1 - \frac{\epsilon}{2})^m
\]

\[
< e^{-\frac{m\epsilon}{2}} \quad \text{; (using } 1 - x < e^{-x})
\]

\[
= e^{-\frac{\epsilon}{2} \cdot \frac{m}{2}} \ln \frac{\epsilon}{2}
\]

\[
= \frac{\delta}{2}
\]

Also, \( Pr(\text{all } m \text{ I.I.D. examples do not hit } [b-,b]) < \frac{\delta}{2} \), by the same argument.

Then it follows that \( Pr(\text{sample hits both } [a,a+] \text{ and } [b-,b]) \geq 1 - \delta \).

In this case, \( err(hyp) = Pr_D[h(x) \neq c(x)] < \epsilon \) is true since:

\[
err(hyp) = Pr_D[a, x^+_{min}] + Pr_D[x^+_{max}, b] \quad (x^+_{min} \text{ is the smallest } \oplus \text{ example, } x^+_{max} \text{ is the largest})
\]

\[
\leq Pr[a, a+] + Pr[b-, b]
\]

\[
= \frac{\epsilon}{2} + \frac{\epsilon}{2}
\]

\[
= \epsilon
\]

NB: The magnitude of \( \delta \) is usually exponentially small, whereas the magnitude of \( \epsilon \) is usually small only in a polynomial sense.

3 Learning Over Logical Domains

Examples are elements of \( \{0,1\}^n \) (i.e. a bit string of length \( n \))

We have \( n \) Boolean variables \( x_1, \ldots, x_n \)

\( x_i \) takes values in \( \{0,1\} \)

E.g.

011010111 ⊕
110111011 ⊕
010110111 ⊗
000111000 ⊗

Hypothesis A conjunction of literals (i.e. a variable or a negated variable)

E.g. \( x_1 \land \neg x_3 \land x_7 \)

Observations For \( \oplus \) examples:

1. The set of literals that are true (=1) in all \( \oplus \) examples include all literals that are in the true concept \( c \).

2. If a literal is false in any \( \oplus \) example, then it is surely not in the true concept.
3.1 The Elimination Algorithm

1. Initially, hypothesis \( = x_1 \land \ldots \land x_n \land \lnot x_1 \land \ldots \land \lnot x_n \)

2. Take \( m = \frac{2n}{\epsilon} \ln \frac{2n}{\delta} \) I.I.D. examples
   - ignore \( \ominus \) examples
   - remove any literal which is false in any positive example

**Theorem:** The Elimination Algorithm PAC learns the class of conjunctions.

For any literal \( z \), we define \( P_z = Pr_D( z = 0 \text{ and true concept } c = 1) \)

*Example:* \( c = x_1 \land \lnot x_3 \land x_7 \). \( P_{x_1} = 0 \), since if \( x_1 \) is false in the example, then \( c \) is false also.

(This holds for any \& all literals in the true concept \( c \))

**Definition:** A literal \( z \) is a *bad* literal if \( P_z > \frac{\epsilon}{2n} \)

The theorem follows from the next two lemmas.

**Lemma 1:** If we remove all bad literals (or the hypothesis does not include any), then \( err(hyp) < \epsilon \)

**Proof:** Notice that by part 1 of the Observations (previous lecture), there is only one type of error - that is error due to including unnecessary literal(s) in the hypothesis. We have:

\[
err(hyp) = Pr[\text{some unnecessary literal(s) in hyp = 0 AND } c(x) = 1]
\]

We use a simple Union Bound:

\[
\sum_{z \in \text{hyp}} P_z \leq \frac{2n \cdot \epsilon}{2n} = \epsilon
\]

**Lemma 2:** With probability at least \( 1 - \delta \) all bad literals are removed by the algorithm.

**Proof:** If a literal \( z \) is bad, then:

\[
Pr[z \text{ is not removed after 1 example}] = 1 - P_z \leq 1 - \frac{\epsilon}{2n}
\]

\[
Pr[z \text{ is not removed after m examples}] \leq (1 - \frac{\epsilon}{2n})^m \leq e^{-\frac{m \cdot \epsilon}{2n}} \leq \frac{\delta}{2n}
\]

\[
Pr[\text{there is a bad literal but we don’t remove it}] \leq 2n \frac{\delta}{2n} = \delta
\]