Mistake Bounded Learning and the Perceptron Algorithm

These notes are slightly edited from previous scribe notes (in Spring 2006) taken by Andrew Fox.

1 Mistake-Bound Learning

Mistake-bound learning can be described in terms of playing an infinite learning game as follows:

1. An adversary chooses some example $x_i$ and shows it to the learner
2. The learner tries to predict the label $\hat{l}_i$ of the example $x_i$
3. The adversary reveals the true label $l_i$ to the learner

**Definition** Algorithm A learns the concept class $C$ with mistake bound $M$ if for any $c \in C$ and any sequence of examples, it makes at most $M$ mistakes.

**Example: A Mistake-Bound Version of the Elimination Algorithm for Learning Conjunctions**

We initialize our hypothesis to $x_1 \land \overline{x}_1 \land \ldots \land x_n \land \overline{x}_n$

Whenever learner makes a mistake (i.e. $\hat{l}_i = -$ but $l_i = +$) it removes the contradicted literals from the hypothesis conjunction.

**Note:** We begin with $2n$ literals
- We remove $n$ on the first mistake
- We remove $\geq 1$ on each mistake thereafter
- Hence $M \leq n + 1$

**Theorem:** If $C$ is learnable with mistake bound $M$, then it is efficiently PAC learnable using $m$ examples, where $m$ is defined below.

**NB:** If we can force the online mistake bound learner to make $M$ mistakes, then its final hypothesis $h$ must have $\text{error}(h) = 0$. It is, however, not always possible to do this in the general case.

**Proof:**

We can use the Online-Learner algorithm as a subroutine to construct a PAC-Learner algorithm as follows:

**PAC-Learner:**

for $i = 1$ to $\infty$

* pick $m_i = \frac{1}{\epsilon} \ln \frac{2^i}{\delta}$ examples
* show examples in sequence to the Online-Learner subroutine
* if the online learner makes a mistake, then go to next iteration
* else output the current hypothesis

If this algorithm reaches iteration \( i = (M + 1) \) then it is guaranteed to stop at this iteration and the current hypothesis will have zero error.

Suppose the PAC-Learner algorithm (above) stops at iteration \( i = 3 \). What can we conclude?

If \( err(hyp) > \epsilon \) then \( Pr[hyp \text{ is consistent}] \leq (1 - \epsilon)^{m_i} = e^{-\ln \frac{\delta}{2^i}} = \frac{\delta}{2^i} \)

The algorithm fails if the output hypothesis \( h \) has \( err(h) > \epsilon \). By applying a Union Bound we can see that:

\[
Pr[\text{PAC-Learner fails}] \leq \sum_{i=1}^{\infty} Pr[\text{output hyp with err} > \epsilon \text{ in the ith iteration}]
\]
\[
\leq \sum_{i=1}^{\infty} \frac{\delta}{2^i} = \delta \sum_{i=1}^{\infty} \frac{1}{2^i} \leq \delta
\]

Then:

\[
m \leq \sum_{i=1}^{M+1} m_i
\]
\[
m_i = \frac{1}{\epsilon} \left( i \ln 2 + \ln \frac{1}{\delta} \right)
\]
\[
m = O \left( \frac{M^2}{\epsilon} + \frac{M}{\epsilon \ln 1/\delta} \right)
\]

Notice however that this algorithm is quite wasteful, since we throw away many examples each time a mistake occurs. Several other variants exist in the literature that give better bounds.

2 Perceptron Algorithm

Examples are in \( \mathbb{R}^n \) (each example has \( n \) features)

A Hypothesis is given by a vector \( \bar{w} \in \mathbb{R}^n \) and \( \theta \in \mathbb{R} \); \( h = (\bar{w}, \theta) \)

\[
h(x) = 1 \iff \sum_{i=1}^{n} w_i x_i \geq \theta
\]
\[
h(x) = -1 \iff \sum_{i=1}^{n} w_i x_i < \theta
\]

If we assume that \( \theta = 0 \) and that labels are denoted by \( l \in \{+1, -1\} \) where +1 signifies a positive example and −1 signifies a negative example, then the perceptron algorithm can be described
as follows:

**Perceptron-Algorithm:**
* Init $\vec{w} = 0$
* Repeat:
  - get example $(\vec{x}, l)$
  - classify example using $\vec{w}$ (as above)
  - if a mistake is made, $\vec{w} \leftarrow \vec{w} + l\vec{x}$

**Assumptions on the Sequence of Examples:**
1. Consider any sequence of examples such that $\exists w^*_{\text{opt}}$ and $\exists \gamma > 0$ and for any example $\vec{x}$ in the sequence: $l(w^*_{\text{opt}} \cdot \vec{x}) = l \sum_i w_{\text{opt},i} x_i \geq \gamma$

2. For all $\vec{x}$ in the sequence, $\|\vec{x}\| = \sqrt{\sum_i x_i^2} \leq R$

Note that the quantity $l(\vec{w} \cdot \vec{x})$ is positive if $\vec{w}$ classifies $\vec{x}$ correctly, and negative if $\vec{w}$ misclassifies $\vec{x}$.

**Theorem** The Perceptron makes $\leq \left( \frac{R}{\gamma} \right)^2 \|w^*_{\text{opt}}\|^2$ mistakes on any sequence of examples that satisfies the assumptions.

**Proof** Consider $(\vec{w}^t \cdot w^*_{\text{opt}})$ such that $\vec{w}^t = \vec{w}^{t-1} + l\vec{x}$ is the learner’s weight vector after the update. We have

$(\vec{w}^t \cdot w^*_{\text{opt}}) = (\vec{w}^{t-1} + l\vec{x}) \cdot w^*_{\text{opt}} = \vec{w}^{t-1} \cdot w^*_{\text{opt}} + l(\vec{x} \cdot w^*_{\text{opt}}) \geq \vec{w}^{t-1} \cdot w^*_{\text{opt}} + \gamma$

Then at each step $t \to t + 1$, $\vec{w}^t$ should get closer and closer to $w^*_{\text{opt}}$.

**Note:** $\vec{w}^0 \cdot w^*_{\text{opt}} = 0 \implies \vec{w}^t \cdot w^*_{\text{opt}} \geq t \cdot \gamma$

We also have: $\|\vec{w}^t\|^2 = \|\vec{w}^{t-1} + l\vec{x}\|^2$

$= \|\vec{w}^{t-1}\|^2 + l^2 \|\vec{x}\|^2 + 2l(\vec{w}^{t-1} \cdot \vec{x})$  [Note: $l^2 \equiv 1$]

$\leq \|\vec{w}^{t-1}\|^2 + R^2$ since $l(\vec{w}^{t-1} \cdot \vec{x}) < 0$ because algorithm made a mistake

Since $\|\vec{w}^0\|^2 = 0$, this implies $\implies \|\vec{w}^t\|^2 \leq tR^2$

Then:

$t \gamma \leq \vec{w}^t \cdot w^*_{\text{opt}} \leq \|\vec{w}^t\| \cdot \|w^*_{\text{opt}}\| \leq \sqrt{tR^2} \cdot \|w^*_{\text{opt}}\|$  

We thus get that $\sqrt{t} \leq \frac{R}{\gamma} \|w^*_{\text{opt}}\|$ and $t \leq \left( \frac{R}{\gamma} \right)^2 \|w^*_{\text{opt}}\|^2$ as required.