Lecture 4: The Perceptron Algorithm (+ continuing on Winnow)

Recap from last time

• Winnow algorithm for learning a k-of-r function: e.g., \( x_3 + x_9 + x_{10} + x_{12} \geq 2 \).
• \( h(x) \): predict \text{pos} iff \( w_1 x_1 + ... + w_n x_n \geq n \).
• Initialize \( w_i = 1 \) for all \( i \).
  - Mistake on pos: \( w_i \leftarrow 2w_i \) for all \( x_i = 1 \).
  - Mistake on neg: \( w_i \leftarrow 0 \) for all \( x_i = 1 \).
• Thm: Winnow makes at most \( O(r \log n) \) mistakes.

• Analysis:
  • Each m.o.p. adds at least \( k \) relevant chips, and each m.o.n removes at most \( k-1 \) relevant chips. At most \( r(1/\epsilon) \log n \) relevant chips total.
  • Each m.o.n. removes almost as much total weight as each m.o.p. adds. Can make \( (1+1/(2k)) \) m.o.n. for every m.o.p. ⇒ Mistake bound \( O((r/\epsilon) \log n) \).

Recap from last time

• Winnow algorithm for learning a disjunction of \( r \) out of \( n \) variables. e.g. \( f(x) = x_3 \lor x_9 \lor x_{12} \).
• \( h(x) \): predict \text{pos} iff \( w_1 x_1 + ... + w_n x_n \geq n \).
• Initialize \( w_i = 1 \) for all \( i \).
  - Mistake on pos: \( w_i \leftarrow 2w_i \) for all \( x_i = 1 \).
  - Mistake on neg: \( w_i \leftarrow 0 \) for all \( x_i = 1 \).
• Thm: Winnow makes at most \( O(r \log n) \) mistakes.

Analysis:

• Each m.o.p. adds at least \( k \) relevant chips, and each m.o.n removes at most \( k-1 \) relevant chips. At most \( r(1/\epsilon) \log n \) relevant chips total.

How about learning general LTFs?

E.g., \( 4x_3 - 2x_9 + 5x_{10} + x_{12} \geq 3 \).

Will look at two algorithms today, each with different types of guarantees:

• Winnow (same as before)
• Perceptron
Winnow for general LTFs

E.g., $4x_3 - 2x_9 + 5x_{10} + x_{12} \geq 3$.

- First, add variable $y_i = 1 - x_i$ so can assume all weights positive.

E.g., $4x_3 + 2y_9 + 5x_{10} + x_{12} \geq 5$.

- Also conceptually scale so that all weights $w_i^*$ of target are integers (not needed but easier to think about)

• Idea: suppose we made $W$ copies of each variable, where $W = w_1^* + \ldots + w_n^*$.

• Then this is just a "$w_0^*$ out of $W$" function!

E.g., $4x_3 + 2y_9 + 5x_{10} + x_{12} \geq 5$.

• So, Winnow makes $O(W^2 \log(Wn))$ mistakes.

• And here is a cool thing: this is equivalent to just initializing each $w_i$ to $W$ and using threshold of $nW$. But that is same as original Winnow!

Winnow for general LTFs

More generally, can show the following (will do the analysis on hwk2):

Suppose $\exists w^*$ s.t.:

- $w^* \cdot x \geq c$ on positive $x$,
- $w^* \cdot x \leq -c - \gamma$ on negative $x$.

Then mistake bound is

- $O((L_1(w^*)/\gamma)^2 \log n)$

Multiply by $L_\infty(X)$ if examples not in $[0,1]$.

Perceptron algorithm

An even older and simpler algorithm, with a bound of a different form.

Suppose $\exists w^*$ s.t.:

- $w^* \cdot x \geq \gamma$ on positive $x$,
- $w^* \cdot x \leq -\gamma$ on negative $x$.

Then mistake bound is

- $O((L_2(w^*)L_2(x)/\gamma)^2)$

L_2 margin of examples

Thm: Suppose data is consistent with some LTF $w^* \cdot x > 0$, where $||w^*|| = 1$ and

$$\gamma = \min_x |w^* \cdot x|/||x||$$

Then # mistakes $\leq 1/\gamma^2$.

Algorithm:
Initialize $w=0$. Use $w \cdot x > 0$.

- Mistake on pos: $w \leftarrow w+x$.
- Mistake on neg: $w \leftarrow w-x$.

(Pre-scale examples to be in unit ball)
**Analysis**

Thm: Suppose data is consistent with some LTF $w^* \cdot x > 0$, where $||w^*||=1$ and

$$\gamma = \min_x |w^* \cdot x| \quad \text{(after scaling so all $||x||=1$)}$$

Then # mistakes $\leq \frac{1}{\gamma^2}$.

Proof: consider $|w \cdot w^*|$ and $||w||$

- Each mistake increases $|w \cdot w^*|$ by at least $\gamma$.
  $$(w + x) \cdot w^* = w \cdot w^* + x \cdot w^* \geq w \cdot w^* + \gamma.$$  
  - Each mistake increases $w \cdot w$ by at most 1.
  $$(w + x) \cdot (w + x) = w \cdot w + 2(x \cdot w) + x \cdot x \leq w \cdot w + 1.$$  
    - So, in $M$ mistakes, $\gamma M \leq |w \cdot w^*| \leq ||w|| \leq M^{1/2}$.
    - So, $M \leq 1/\gamma^2$.

**What if no perfect separator?**

In this case, a mistake could cause $|w \cdot w^*|$ to drop.

The $\gamma$-hinge-loss of $w^* = \Sigma_x \max[0, 1 - l(x)(x \cdot w^*)/\gamma]$ (by how much, in units of $\gamma$, would you have to move the points to all be correct by $\gamma$)

Proof: consider $|w \cdot w^*|$ and $||w||$

- Each mistake increases $|w \cdot w^*|$ by at least $\gamma$.
  $$(w + x) \cdot w^* = w \cdot w^* + x \cdot w^* \geq w \cdot w^* + \gamma.$$  
  - Each mistake increases $w \cdot w$ by at most 1.
  $$(w + x) \cdot (w + x) = w \cdot w + 2(x \cdot w) + x \cdot x \leq w \cdot w + 1.$$  
    - So, in $M$ mistakes, $\gamma M \leq |w \cdot w^*| \leq ||w|| \leq M^{1/2}$.
    - So, $M \leq 1/\gamma^2$.

**Kernel functions**

See board...