Learning when there is no perfect predictor

- Hoeffding/Chernoff bounds: minimizing training error will approximately minimize true error: just need $O(1/\epsilon^2)$ samples versus $O(1/\epsilon)$.  
- What about polynomial-time algorithms? Seems harder.  
  - Given data set $S$, finding apx best conjunction is NP-hard.  
  - Can do other things, like minimize hinge-conj, but may be a big gap wrt error rate ("0/1 loss").  
- One way to make progress: make assumptions on the "noise" in the data. E.g., Random Classification Noise model.

Learning from noisy data, intro to SQ model

Avrim Blum  
05/04/20

Learning from Random Classification Noise

- PAC model, target $f \in C$, but assume labels from noisy channel.  
- "noisy" oracle $EX^\eta(f, D)$. $\eta$ is the noise rate. (think $\eta = \frac{1}{2}$)  
  - Example $x$ is drawn from $D$.  
  - With probability $1-\eta$ see label $\ell(x) = f(x)$.  
  - With probability $\eta$ see label $\ell(x) = 1-f(x)$.  
- E.g., if $h$ has non-noisy error $p$, what is the noisy error rate? (If $X_f(h,x) = f(x)$ then $X^\eta(h,x) = f(x)$?)  
  - $p(1-\eta) + (1-p)\eta = \eta + p(1-2\eta)$.  

Learning OR-functions (assume monotone)

- Let's assume noise rate $\eta$ is known.  
- Say $p_1 = P[f(x)=0 \text{ and } x_1=1]$ (if $x_1$ in target then $p_1 = 0$)  
- Any $h$ that includes all $x_i$ such that $p_i=0$ and no $x_i$ such that $p_i > \epsilon/n$ is good. (e.g., think of $f = x_1 \lor x_2 \lor x_3$)  
  - So, just need to estimate $p_1$ to $\pm \frac{\epsilon}{2n}$  
    - Rewrite as $p_1 = P[f(x)=0 | x_1=1] \times P[x_1=1]$.  
    - 2nd part unaffected by noise (and if tiny, then $p_1$ is small for sure).  
      - Define $q_1$ as 1st part.  
    - Then $P[f(x)=0 | x_1=1] = q_1(1-\eta) = (1-q_1)\eta = q_1(1-2\eta)$.  
    - So, enough to approx LHS to $\pm 0 \left( \frac{\epsilon}{2n} (1 - 2\eta) \right)$.  

Notation

- Use "Pr[...]", for probability with respect to non-noisy distribution.  
- Use "Pr_n[...]", for probability with respect to noisy distribution.
Learning OR-functions (assume monotone)

- If noise rate not known, can estimate with smallest value of $\Pr_{\eta}[(x) = 0 | x_i = 1]$.

$$
\begin{array}{c|c|c}
0 & 0 & \eta \\
0 & 1 & 1 - \eta \\
\end{array}
$$

(e.g., $f = x_1 \vee x_3 \vee x_5$)

Generalizing the algorithm

Basic idea of algorithm was:
- See how can learn in non-noisy model by asking about probabilities of certain events with some "slop".
- Try to learn in noisy model by breaking events into:
  - Parts predictably affected by noise.
  - Parts unaffected by noise.

Let's formalize this in notion of "statistical query" (SQ) algorithm. Will see how to convert any SQ alg to work with noise.

The Statistical Query Model

- No noise.
- Algorithm asks: "what is the probability a labeled example will have property $\chi$? Please tell me up to additive error $\tau$." (e.g., $x_i = 1$ and label is negative)
  - Formally, $\chi : X \times \{0,1\} \to \{0,1\}$. Must be poly-time computable. $\tau \geq 1/poly(\ldots)$.
  - Let $P'_\chi = P_\chi \pm \tau$, where $P_\chi = \Pr_{x \sim D}[\chi(x,f(x)) = 1]$.
  - World responds with $P'_\chi \in [P_\chi - \tau, P_\chi + \tau]$.
  - May repeat poly(… times. Can also ask for unlabeled data. Must output $h$ of error $\leq \epsilon$. No $\delta$ in this model.

Data $\rightarrow$ Alg

The Statistical Query Model

- Many algorithms can be simulated with statistical queries:
  - Perceptron: ask for $E[f(x) : h(x) = f(x)]$ (formally define vector-valued $\chi$ = $f(x)$ if $h(x) = f(x)$, and 0 otherwise. Then divide by $\Pr[h(x) = f(x)]$.)
  - Hill-climbing type algorithms: what is error rate of $h$?
    What would it be if I made this tweak?

Properties of SQ model:

- Can automatically convert to work in presence of classification noise.
- Can give a nice characterization of what can and cannot be learned in it.

Data $\rightarrow$ Alg

SQ-learnable $\Rightarrow$ (PAC+Noise)-learnable

- Given query $\chi$, need to estimate from noisy data. Idea:
  - Break into part predictably affected by noise, and part unaffected.
  - Estimate these parts separately.
  - Can draw fresh examples for each query or estimate many queries from same sample if VCDim of query space is small.

Running example: $\chi(x, l) = 1$ iff $x_i = 1$ and $l = 0$. 
How to estimate $\Pr[\chi(x,f(x))=1]$?

- Let $\text{CLEAN} = \{x : \chi(x,0) = \chi(x,1)\}$
- Let $\text{NOISY} = \{x : \chi(x,0) \neq \chi(x,1)\}$

Now we can write:

- $\Pr[\chi(x,f(x))=1] = \Pr[\chi(x,f(x))=1 \text{ and } x \in \text{CLEAN}] + \Pr[\chi(x,f(x))=1 \text{ and } x \in \text{NOISY}]$.

Step 1: first part is easy to estimate from noisy data (easy to tell if $x \in \text{CLEAN}$).

What about the 2nd part?

Can estimate $\Pr[x \in \text{NOISY}]$.

Also estimate $P_\eta \equiv \Pr[\chi(x,f)=1 \mid x \in \text{NOISY}]$.

Write $P_\eta = P(1-\eta) + (1-P)\eta = \eta + P(1-2\eta)$.

So, $P = (P_\eta - \eta) / (1-2\eta)$.

Just need to estimate $P_\eta$ to additive error $\pm(1-2\eta)$.

If don’t know $\eta$, can have “guess and check” wrapper.

Characterizing what’s learnable using SQ algorithms

- Say that $f,g$ uncorrelated if $\Pr_{x \sim D}[f(x) = g(x)] = \frac{1}{2}$.

Def: the SQ-dimension of a class $C$ wrt $D$ is the size of the largest set $C' \subseteq C$ s.t. for all $f,g \in C'$,

$$\left| \Pr_D[f(x) = g(x)] - \frac{1}{2} \right| < \frac{1}{|C'|}$$

(size of largest set of nearly uncorrelated functions in $C$)

- Theorem 1: if $\text{SQDIM}_D(C) = \text{poly}(n)$ then you can weak-learn $C$ over $D$ by SQ algs. ($\text{error rate} \leq \frac{1}{2} - \frac{1}{\text{poly}(n)}$)

- Theorem 2: if $\text{SQDIM}_D(C) > \text{poly}(n)$ then you can’t weak-learn $C$ over $D$ by SQ algs.

Key tool: Fourier analysis of boolean functions.

Sounds scary but it’s a cool idea!

Let’s think of functions from $\{0,1\}^n \rightarrow \{-1,1\}$.

View function $f$ as a vector of $2^n$ entries:

$$(\sqrt{D[000]}f(000), \sqrt{D[001]}f(001), \ldots, \sqrt{D[x]}f(x), \ldots)$$

What is $(f,f)$? What is $(f,g)$?

What is an orthonormal basis?