Lecture 16

Fast weak learning of monotone functions
Monotone Functions

def partial order ≤

\[ x \leq y \iff \forall i, x_i \leq y_i \]

monotone function f

\[ x \leq y \Rightarrow f(x) \leq f(y) \]

Learning algorithms for the class of monotone functions?

in homework we saw 2 random samples so easily

why is this nontrivial?

we said poly samples is easy

the problem is computation time? poly in what?

but read poly(\log |E|) samples

all monotone functions

there are \( 2^n \) functions, \( \geq 2^{\frac{n}{m}} \) monotone functions

why \( \geq 2^n \) monotone functions?

consider slice functions:

\[ \begin{cases} f=1 \rightarrow \Delta \\ f=0 \rightarrow \nabla \end{cases} \]

\( \geq \) learning needs \( D(2^n) \)

even with queries in PAC model
Today's question: what about learning monotone distributions, on uniform distribution, with queries? Here we will get a very slight "win": All monotone functions have weak agreement with some dictator function.

**Thm** $f$ monotone, $f \in \{\pm 1, x_1, x_2, \ldots, x_n^3\}$ such that $P_x[f(x) = g(x)] \geq \frac{1}{2} + \Omega\left(\frac{1}{n}\right)$

**Pf.**

**Case 1** $f(x)$ has weak agreement with $+1$ or $-1$.
**Case 2** otherwise $P_x[f(x) = 1] \in \left[\frac{1}{4}, \frac{3}{4}\right]$.

First a break, before we prove case 2...

what is another way to think of influence of monotone functions?

- # nodes = $2^n$, # edges = $\frac{n \cdot 2^n}{2}$
- each level has $\binom{n}{j}$ weight $j$ nodes
- monotone \( \Rightarrow \) no blue above any red
- sliding cube in roughly half cuts many edges at many in same direction
- \( I_a(f) = \frac{\text{red-blue edges}}{2^{a-1}} \), \( I_b(f) = \frac{\text{rb edges in } a^{th} \text{ dir}}{2^{n-1}} \)
Recall H.W.:

\[
\text{Then } f \text{ monotone, } \quad \ln f_\infty (f) = \hat{f}(\varepsilon \delta_3) = \Pr[f(x) = \chi_{\varepsilon \delta_3}(x)] - 1
\]

\[\uparrow \quad \text{H.W.} \quad \uparrow \text{Known} \quad \chi_{\varepsilon \delta_3}(x)\]

Plan:

Show \( \ln f_\infty (f) = \Omega(\frac{1}{n}) \)

\[
\Rightarrow \Pr[\hat{f}(x) = X_i] \geq \frac{1}{2} + \ln f_\infty (f) \geq \frac{1}{2} + \Omega(\frac{1}{n})
\]

Will use following tool:

**Canonical Path Argument**

Plan 1) define canonical path for every red-blue pair of nodes (note such a path must cross at least one red-blue edge)

2) show upper bnd on # of c.p.l.s passing through any edge (in particular, any red-blue edge)

3) conclude lower bnd. on # of red-blue edges
Part 1 of plan:

\[ \text{def. } V(x, y) \text{ s.t. } x \text{ red } \& y \text{ blue} \]

"canonical path from } x \text{ to } y" \text{ is: }

scan bits left to right, flipping where needed

each flip \( \Rightarrow \) step in path

\[
\begin{align*}
\text{Example} & & \text{direction} & & 1 & & 2 & & 3 & & 4 \\
x & = & -1 & & +1 & & +1 & & +1 & & +1 \\
w & = & 5 & & +1 & & +1 & & +1 & & +1 \\
z & = & +1 & & -1 & & +1 & & +1 & & 5 \\
y & = & +1 & & -1 & & +1 & & -1 & & 1
\end{align*}
\]

\( x \Rightarrow w \Rightarrow z \Rightarrow y \)

each step is Hamming distance 1

How many red-blue pairs have canonical paths?

Recall \( P \{ f(N) = 1 \} \approx \left[ \frac{1}{4}, \frac{3}{4} \right] \)

\( \text{# paths } \geq \frac{1}{4} \cdot 2^n \cdot \frac{1}{4} \cdot 2^n = \frac{1}{16} \cdot 2^{2n} \)
Part 2 of plan:

For any (red-blue) edge $e$, how many $x,y$ pairs can cross it with canonical $x,y$-path?

Main point:
- all canonical paths crossing $u, u^{@x}$ agree on $y_1...y_{i-1}$ + $x_i...x_n$

Part 3 of plan:

$$(\# \text{ red-blue edges} \leq \max \# \text{ canonical paths} \text{ that use it}) \geq \# \text{ red-blue canonical paths}$$

So

$$\# \text{ red-blue edges} \geq \frac{1}{16} 2^{2n} = \frac{1}{16} 2^n$$

So $\exists i$ s.t. $\geq \frac{2^n}{16} \cdot \frac{1}{n}$ red-blue edges in direction $i$
so \[
\inf_x (f) = \frac{2^n}{16n} = \frac{1}{8n} = \hat{f}(\bar{x}3) = 2\Pr[f(x) = x_3] - 1
\]
\[
\text{total # edges}
\text{in dis. 1}
\]
\[
\therefore \Pr[f(x) = x_3] \geq \frac{1}{2} + \frac{1}{16n}
\]

Canonical Path argument also used in
- routing
- expansion/conductance of hypercube/other Markov Chains

What good is weak learning?

unclear
here only uniform distribution
if can learn in all distributions,
can do much more

(next result does not apply to monotone
function learning... i.e. in agreement
in particular, this weak notion of learning i.e. const \( \approx \frac{1}{n} \) agreement
probably doesn't give anything for stronger learning)
Weak vs. Strong Learning

Def. Algorithm A weakly "PAC learns" concept class C
if \( \forall c \in C \) \( \forall \delta > 0 \)
\( \forall \epsilon, \delta > 0 \) \( (\delta = \frac{1}{4} \text{ or } \frac{1}{4} \text{ doesn't affect}) \)
with prob \( \geq 1 - \delta \)
given examples of C
A outputs h s.t. \( \Pr_{x \leftarrow \mathcal{D}} \left[ h(x) \neq c(x) \right] \leq \frac{\epsilon}{2} - \frac{\delta}{2} \)

It was conjectured that distribution free weak learning
was really weaker but surprise!

Can "boost" a weak learner

Then if C can be weakly learned on
any dist \( \mathcal{D} \) then C can be
(strongly) learned.
Applications

1) "Theoretical"
   - Unit test Algorithms for poly term DNF weight w- poly threshold fcn's
     \[ \text{low degree only doesn't work well} \]
     \[ \text{(Boosting + KM)} \]
   - Ave case vs. worst case

2) practical - Boosting
   - Freund-Schapire

Good & Bad Ideas

1) simulate weak learner several times on
   same distribution & take majority answer
   \[ \text{or} \]
   best answer
   gives better confidence
   but doesn't reduce error, what if always get same answer?

2) fill in examples on which current hypothesis
does well & run weak learner on part where you
do badly.

Problem: given a new example, how do you
know which section it is in?
3) Keep some samples on which you are ok
always use majority vote on all previous hypotheses
to predict value of new samples

history: Schapire, Freund-Schapire, Impagliazzo-
Servedio, Klivans

Filtering Procedures
- decide which samples to keep, which to throw out
- samples on which so far you guess correctly ← need for checking
  future hypotheses
  incorrectly ← need to improve on these