Lecture 16

fast weak learning of monotone functions

Mon (1)
Sp 2014

Monotone Functions

<u>def</u> partial order $\leq$

$$x \leq y \quad \text{iff } \forall i \quad x_i \leq y_i$$

monotone function f

$$x \leq y \Rightarrow f(x) \leq f(y)$$

Learning algorithms for the <u>class</u> of monotone functions?

in homework we saw $2^{O(\sqrt{n})}$ random samples suffice for uniform distribution

why is this nontrivial?

we said poly samples is easy, the problem is computation time? poly in what?

but need $\text{poly}(\log |C|)$ samples
↑ all monotone fctns

there are 2^{2^n} fctns, $\geq 2^{2^n/\sqrt{n}}$ monotone fctns

why $\geq 2^{2^n}$ monotone fctns?

consider slice fctns:

$f = 1 \rightarrow$

$f = 0 \rightarrow$

$\binom{n}{\frac{n}{2}} \approx 2^n/\sqrt{n}$ pts in middle row can be set in all possible ways w/o violating monotonicity

a hard distribution: Uniform on middle row ⟹

$\Rightarrow$ learning needs $\Omega\left(\frac{2^n}{\sqrt{n}}\right)$ even with queries in PAC model

Mon ②
Sp2014

Today's question:
what about learning monotone distributions,
on uniform distribution,
<u>with</u> queries?

here we will get a very slight "win":

All monotone fctns have weak agreement $\left(\frac{1}{2}+\theta\left(\frac{1}{n}\right)\right)$ with <u>some</u> dictator fctn.

slightly better than random guess

(can get $\frac{1}{2}+\theta\left(\frac{1}{\sqrt{n}}\right)$ with majority + dictators)

<u>Thm</u> $\forall f$ monotone, $\exists\, g \in \{\pm 1, x_1, x_2, \ldots, x_n\} = S$
s.t. $\Pr_x[f(x)=g(x)] \geq \frac{1}{2}+\Omega\left(\frac{1}{n}\right)$

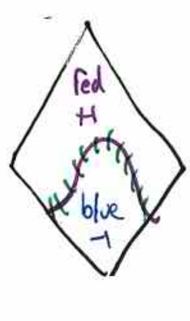
Why does this give weak learning algorithm? estimate agreement of f with all members of S + output member with max agreement

<u>Pf.</u>

<u>Case 1</u> $f(x)$ has weak agreement with $+1$ or -1

<u>Case 2</u> otherwise $\Pr[f(x)=1] \in \left[\frac{1}{4}, \frac{3}{4}\right]$

First a break,
before we prove case 2 ...
what is another way to think of influence of monotone fctns?



- $\#$ 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
- slicing cube in roughly half cuts many edges + many in same direction
- $\mathrm{Inf}(f) = \frac{\#\text{red-blue edges}}{2^{n-1}}$, $\mathrm{Inf}_i(f) = \frac{\#\text{rb edges in } i^{th} \text{dir}}{2^{n-1}}$

Mon ③
Sp 2014

recall H.W.:

Thm f monotone

$$\mathrm{Inf}_i(f) \underset{\substack{\uparrow \\ \text{H.W.}}}{=} \hat{f}(\{i\}) \underset{\substack{\uparrow \\ \text{Known}}}{\equiv} 2\Pr[f(x) = \underbrace{\chi_{\{i\}}}_{x_i}(x)] - 1$$

Plan:

Show $\mathrm{Inf}_i(f) \geq \Omega(\frac{1}{n})$

$$\Rightarrow \Pr[f(x) = x_i] \geq \frac{1}{2} + \frac{\mathrm{Inf}_i(f)}{2}$$
$$\geq \frac{1}{2} + \Omega(\frac{1}{n})$$

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.'s passing through any edge (in particular, any red-blue edge)

3) conclude lower bnd. on # of red-blue edges

Part 1 of plan:

<u>def</u>. $\forall (x,y)$ s.t. x red & y blue

"canonical path from x to y" is:

scan bits left to right, flipping where needed

each flip $\rightsquigarrow$ step in path

<u>example</u>

	direction 1	2	3	4
$x =$	-1	$+1$	$+1$	$+1$
$w =$	$+1$	$+1$	$+1$	$+1$
$z =$	$+1$	-1	$+1$	$+1$
$y =$	$+1$	-1	$+$	-1

$x \to w \to z \to y$
each step is Hamming distance 1

How many red-blue x,y pairs have canonical paths?

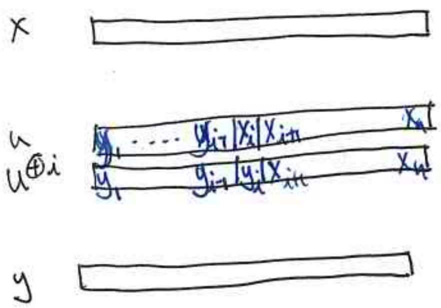
recall, $\Pr[f(x)=1] \in [\frac{1}{4}, \frac{3}{4}]$

$$\#\text{ paths} \geq \frac{1}{4}\cdot 2^n \cdot \frac{1}{4}\cdot 2^n = \frac{1}{16}\cdot 2^{2n}$$

Part 2 of plan:

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

$e = (u, u^{\oplus i})$
red blue



edge in i^{th} direction

$\underbrace{y_1 \cdots y_{i-1}}_{\leq 2^{i-1} \text{ settings for prefix of } x}$ x_i $x_{i+1} \cdots x_n$

$y_1 \cdots y_{i-1}$ y_i $\underbrace{x_{i+1} \cdots x_n}_{\leq 2^{n-i} \text{ settings for suffix of } y}$

$\therefore \leq 2^n$ total settings of prefix x, suffix y consistent with this edge

Main point: all canonical paths crossing $u, u^{\oplus i}$ agree on $y_1 \cdots y_{i-1}$ + $x_{i+1} \cdots x_n$

Part 3 of plan:

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

since each uses ≥ 1 red-blue edge

So

$$\text{\# red-blue edges} \geq \frac{\overset{\text{l.b. on \# r-b pairs}}{\frac{1}{16} 2^{2n}}}{\underset{\text{u.b. on \# canonical paths crossing any edge}}{2^n}} = \frac{1}{16} \cdot 2^n$$

so $\exists\, i$ s.t. $\geq \frac{2^n}{16} \cdot \frac{1}{n}$ red-blue edges in direction i

so $$\mathrm{Inf}_i(f) \ge \frac{\frac{2^n}{16n}}{\underset{\substack{\uparrow\\ \text{total \# edges}\\ \text{in dir } i}}{2^{n-1}}} = \frac{1}{8n} = \hat{f}(\{i\}) = 2\Pr[f(x)=x_i]-1$$

$$\therefore \Pr[f(x)=x_i] \ge \frac{1}{2}+\frac{1}{16\cdot n}$$

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...

in particular, this weak notion of learning provably doesn't give anything for stronger learning)

i.e. $\frac{1}{n}$ agreement

i.e. const $> \frac{1}{2}$ agreement

Weak vs. Strong Learning

Def. Algorithm $\mathcal{A}$ weakly "PAC learns" concept class $\mathcal{C}$

if $\forall\ c \in \mathcal{C}$ & $\forall$ dists $\mathcal{D}$ $\quad \exists\ \gamma > 0$

$\forall\ \cancel{\varepsilon}, \delta > 0$ $\quad$ ($\delta = \frac{1}{4}$ or $\frac{1}{n^2}$ doesn't affect)

with prob $\geq 1-\delta$

given examples of c

$\mathcal{A}$ outputs h s.t. $\Pr_{\mathcal{D}}[h(x) \neq c(x)] \leq \cancel{\varepsilon}\ \frac{1}{2} - \frac{\gamma}{2}$

advantage

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

Can "boost" a weak learner

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

Applications

1) "theoretical"

- Unif dist Algorithms for poly term DNF
  weight w - poly threshhold fctns } low degree alg doesn't work well

  ∴ (Boosting + KM)

- Ave case vs. worst case

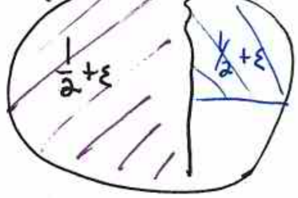
2) practical - Boosting
   Freund-Schapire

Good & Bad Ideas

1) simulate weak learner several times on same distribution & take majority answer
   -or-
   best answer

   gives better confidence
   but doesn't reduce error, what if always get same answer?

2) filter out 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