Computational Learning Theory

June 15, 2025

Maths

Hoeffding's inequality:

$X_1, ..., X_m$ m indep RVs in $[0,1]$, $\overline{X} := \frac{1}{m}\sum_{i=1}^{m} X_i$, $\mu := \mathbb{E}[\overline{X}]$. $\forall t \geq 0$:

$$\mathbb{P}\left[\left|\overline{X} - \mu\right| \geq t\right] \leq 2\exp(-2mt^2)$$

Chernoff-Hoeffding bound (from S1):

$X_1, ..., X_m$ m indep RVs in $[0,1]$, $X := \sum_{i=1}^{m} X_i$, $\mu := \mathbb{E}[X] = \sum_i \mathbb{E}[X_i]$. $\forall \alpha \geq 0$:

$$\mathbb{P}\left[|X - \mu| \geq \alpha\right] \leq 2\exp(-2t^2/m)$$

[proof by Hoeffding bound with $t := \alpha/m$.

Chernoff bound

$X_1, ..., X_m$ m indep RVs in $\{0,1\}$, $X := \sum_{i=1}^{m} X_i$, $\mu := \mathbb{E}[X] = \sum_i \mathbb{E}[X_i]$. $\forall \delta \in [0,1]$:

$$\mathbb{P}\left[X \leq (1-\delta)\mu\right] \leq \exp(-\delta^2\mu/2)$$

$$\mathbb{P}\left[X \leq (1+\delta)\mu\right] \leq \exp(-\delta^2\mu/3)$$

$1 - x \leq e^{-x}$

symmetrization argument

Classes

Conjunctions

k-CNF: clasuses of length exactly k

3-term DNF: formulae with exactly three 3 terms (which are conjunctions in themselves)

1 PAC Learning

$\mathrm{err}(h; c, D) := \mathbb{P}_{x \sim D}[h(x) \neq c(x)]$

instance space $X := \bigcup_{\geq 1} X_n$, where $X_n = \{0,1\}^n$ or $\mathbb{R}^n$ (essentially spaces of dimension n)

concept class $C := \{c : X \to \{0,1\}\}$, which we may split as $C_n := \{c : X_n \to \{0,1\}\}$

distribution D over X, we have an oracle $\texttt{EX}(c, D)$ that returns pairs $(x, c(x))$ where $x \sim D$

representation scheme for C is an onto function $R : \Sigma^* \to C$, from the set of strings on some "finite" alphabet (which may include $\mathbb{R}$ if necessary, treating 1 number as a unit character)

$\mathrm{size}(c) := \min_{\sigma : R(\sigma) = c}\{\mathrm{size}(c)\}$

polynomially evaluatable Hypothesis class: $H := \bigcup_{n \geq 1} H_n$, where H_n is the set of hypotheses over X_n, where there exists an algorithm that outputs $h(x)$ on input $x \in X_n, h \in H_n$ in time poly in $n, \mathrm{size}(h)$

PAC learning:

C is PAC learnable using hypothesis class H if:

- $\exists$an alg L st $\forall n \in \mathbb{N}\ \forall c \in C_n, \forall$dist D over X_n, $\forall \varepsilon \in (0, 1/2), \forall \delta \in (0, 1/2)$
- given inputs $n, \mathrm{size}(c), \varepsilon, \delta$, access to $\texttt{EX}(c, D)$
- L outputs a hypothesis $h \in H_n$ that satisfies $\mathrm{err}(h) \leq \varepsilon$ with probability $\geq 1 - \delta$ (prob over samples from the oracle and internal randomness)
- where the number of calls to $\texttt{EX}(c, D)$ is bounded by a poly in $1/\varepsilon$,$1/\delta, n$ and $\mathrm{size}(c)$
- and$H$ is polynomially evaluatable
- it is **efficiently** PAC learnable (using H) if the running time of L is bounded by a poly in $1/\varepsilon$,$1/\delta, n$ and $\mathrm{size}(c)$

In practise, $\mathrm{size}(c)$ is generally ignored, as most concept classes have a bounded representation/ the algorithm does not depend on it.

2 Consistent learning

an algorithm L is a **consistent learner** for a concept class C w/ hypothesis class H st:

- $\forall n \geq 1, \forall c \in C_n, \forall m \geq 1$,
- given as input m examples $(x_i, c(x_i))$,

- L outputs $h \in H_n$ st $\forall i\ h(x_i) = c(x_i)$.

L is an **efficient** consistent learner if the running time of L is polynomial in $n, \text{size}(c), m$

Theorem. Occam's razor:

Given C, H, L a consistent learner of C using H:

$\forall n \geq 1, \forall c \in C_n, \forall D$ over X_n, $\forall \varepsilon \in (0, 1/2), \forall \delta \in (0, 1/2)$, if L is given a sample of size m from $\texttt{EX}(c, D)$ st

$$m \geq \frac{1}{\varepsilon}\left(\log |H_n| + \log \frac{1}{\delta}\right)$$

then L is guaranteed to output a hypothesis $h \in H_n$ with prob $\geq 1-\delta$ st $\text{err}(h) \leq \varepsilon$.

C is then efficently PAC learnable if L is an efficient *consistent learner, $\log |H_n|$ is poly in $n, \text{size}(c)$, and H is polynomially evaluatable.*

Proof. fix n, c, D. Let $h \in H_n$ be "bad" if $\text{err}(h) \geq \varepsilon$.

Draw m independent samples from D, call this S. Let A_h be the event that hypothesis h is consistent on S.

Thus, if h is bad, then $\mathbb{P}[A_h] \leq (1-\varepsilon)^m \leq \exp(-\varepsilon m)$

Let $\mathcal{E} = \bigcup_{h \in H_n : h \text{ bad}} A_h$, so $\mathbb{P}[\mathcal{E}] \leq \sum_{h \text{ bad}} \mathbb{P}(A_h) \leq |H_n| \cdot \exp(-\varepsilon m) \leq \delta$ by choice of m

And the failure of L is a subset of $\mathcal{E}$. □

Variation for $H_{n,m}$: replace H_n by $H_{n,m}$, so the hypothesis also depends on the data size. The bound for PAC learning is now: $\log |H_{n,m}|$ bounded by $\text{poly}(n, \text{size}(c), 1/\varepsilon, 1/\delta) \cdot m^\beta$ for $\beta < 1$ [proof v. similar, uses fact that m can just be divided out of the bound, but now we have $m \geq (2/\varepsilon \cdot \text{poly}(n, \text{size}(c), 1/\varepsilon, 1/\delta))^{1/1-\beta}$, so depends on β too.]

Hypothesis class sizes for common classes:

Conjunctions: size 3^n: each variable can be pos, neg or not appear (if we have both pos+neg, then this is the false conjunction - so need $+1$)

3-term DNF: $\leq 2^{6n}$: each conjunction can be represented by length $2n$ bitstring, need 3 of these, then there are 2^{6n} possible bit strings to represent (some of which are the same)

3-CNF: $2^{c \cdot n^3}$: each cnf formula can be represented by a length $(2n)^3$ (no. of possible clauses, 1/0 for whether in formula), also can show there are $\geq 2^{Cn^3}$ clauses, so bound is tight enough.

3 VC dimension

intuiton: largest set where any pattern can be represented by C (e.g. below this size, any noise could be matched by a different concept)

proving VCD: need 1 set of size d that you can demonstrate all dichotomies on, and for all sets of size $> d$, need to show 1dichotomy fails.

Given instance space X, concept class $C = \{c : X \to \{0, 1\}\}$

for a finite subset $S \subseteq X$, $\Pi_C(S) := \{c|_S : c \in C\}$ is the set of concepts, restricted to just S.

$\Pi_C(S) = \{(c(x_1), ..., c(x_n) : c \in C\}$, as $S = \{x_1, ..., x_n\}$(as finite)

so $|\Pi_C(S)| \leq 2^m$ as c boolean.

$S \subseteq X$ is **shattered** by C if $|\Pi_C(S)| = 2^{|S|}$, so all poss. dichotomies over S can be realised by C - equiv, for any subset of S, there is a $c \in C$ st $\{x : c(x) = 1\} = S$

Vapnik Chervonenkis dimension: $\mathrm{VCD}(C) :=$ the cardinality d of the largest finite set S shattered by C. If C shatters arbitrarily large finite sets, then $\mathrm{VCD}(C) = \infty$

Growth function: for C over X, $\forall m \in \mathbb{N}$,

$$\Pi_C(m) := \max\{|\Pi_C| : S \subseteq X, |S| = m\}$$

for $m \leq \mathrm{VCD}(C), \Pi_C(m) = 2^m$ - how does it behave above the VC dimension?

Sauer's lemma: Given $d := \mathrm{VCD}(C)$, for $m \geq d$, $\Pi_C(m) \leq \left(\frac{me}{d}\right)^d$ (i.e. $O(m^d)$)

Examples:

- intervals in $\mathbb{R}$: 2
- hyper-rectangles: $2n$
- linear halfspaces $n + 1$ (see a sheet)

Sample complexity upper bound

Theorem. *Given $C, H \supseteq C$ with $\mathrm{VCD}(H) = d$, L a consistent learner of C using H:*

$\forall n \geq 1, \forall c \in C_n, \forall D$ over X_n, $\forall \varepsilon > 0, \forall \delta \in (0, 1/2)$, if L is given a sample of size m from $\mathtt{EX}(c, D)$ st

$$m \geq \kappa_0 \left(\frac{1}{\varepsilon} \log \frac{1}{\delta} + \frac{\mathrm{VCD}(H)}{\varepsilon} \log \frac{1}{\varepsilon}\right)$$

then L is guaranteed to output a hypothesis $h \in H_n$ with prob $\geq 1-\delta$ st $\mathrm{err}(h) \leq \varepsilon$. ($\kappa_0$ is a universal constant, irrespective of $n, \varepsilon, \delta, c, D$)

This is <u>efficient</u> PAC learning if L is efficient, and $\mathrm{VCD}(H)$ is polynomial in n.

Proof. ASDF

For notation, $f \oplus g = x \mapsto 1 \iff f(x) \neq g(x)$, otherwise 0.

let $H \oplus c = \{h \oplus c : h \in H\}$, and note that $\mathrm{VCD}(H \oplus c) = \mathrm{VCD}(H)$: for a lower bound, take S of size $\mathrm{VCD}(H)$, and for any dichotomy f, $\exists h$ st $h \oplus c = f$ (by taking $c(x)$ when $f(x) = 0$, otherwise $\neg c(x)$. for an upper bound, $(H \oplus c) \oplus c = H$ because of associativity/commutativity/etc.

Here, $\mathrm{err}(h; c, D) = \mathbb{P}_{x \sim D}[(c \oplus h)(x) = 1]$

$S \subseteq X$ is an ε-net for H' if $\forall h \in H'$ st $\mathbb{P}[h(x) = 1] \geq \varepsilon$ $\exists x \in S$ st $h(x) = 1$

Thus if a finite sample S is an ε-net for $H \oplus c$, it rules out any hypotheses inconsistent with c.

Drawing S of size $2m$:

Draw a sample S_1 size $m$from $\mathtt{EX}(c, D)$. Let A be the event that S_1 is not an ε-net for $H \oplus c$. If A occurs, then $\exists \tilde{h} \in H$ st $(\tilde{h} \oplus c)(S_1) = 0$ and $\mathbb{P}[(\tilde{h} \oplus c)(x) = 1] \geq \varepsilon$.

Fix such a $\tilde{h}$, and draw a sample S_2 of size m. let X_i be the indicator RV for the ith value of S_2 satisfying $(\tilde{h} \oplus c)(x) = 1$. Let $X = \sum_i X_i$, and $\mathbb{E}[X_i] \geq \varepsilon$, so $\mathbb{E}[X] \geq m\varepsilon$

A Chernoff bound gives $\mathbb{P}[X < \varepsilon m/2] \leq \exp(-\varepsilon m/16)$ (with $\delta = 1/2$ for the bound) with the m we have.

Define the event B that given a sample $S = S_1 \cup S_2$, each size m, that $\exists h' \oplus c \in \Pi_{H \oplus c}(S)$ st $|\{x \in S : (h' \oplus c)(x) = 1\}| \geq \varepsilon m/2$ and $(h' \oplus c)(S_1) = 0$.

Thus, $\mathbb{P}[B] = \mathbb{P}[B|A] \cdot \mathbb{P}[A] + \mathbb{P}[B|\neg A] \cdot \mathbb{P}[\neg A] \geq 1/2 \cdot \mathbb{P}[A]$ by above.

Bound on $\mathbb{P}[B]$:

$$\Pi^{\varepsilon}_{H \oplus c}(S) := \{h \oplus c \in \Pi_{H \oplus c}(S) : |\{x \in S : (h \oplus c)(x) = 1\}| \geq \varepsilon m/2\}$$

We can consider S as 1 sample of size $2m$ drawn from D, then partitioned.

For a fixed $h \oplus c \in \Pi^{\varepsilon}_{H \oplus c}(S)$, let $B_{h \oplus c}|S$ be the event that $|\{x \in S_1 : (h \oplus c)(x) = 1\}| = 0$, conditioned on S.

$\mathbb{P}[B_{h \oplus c}|S]$ is equiv to: 2m balls, $r \geq \varepsilon m/2$ are red (1)- if divided into two sets size m, $\mathbb{P}$ first set has no red balls. This is

$$\frac{\binom{m}{r}}{\binom{2m}{r}} \leq \prod_{i=0}^{r-1} \frac{m-i}{2m-i} \leq \frac{1}{2^r} \leq 2^{-\varepsilon m/2}$$

NB. bound is still valid if $r > m$, as then $\mathbb{P}[B_{h \oplus c}|S] = 0$.

$$\begin{aligned}
\mathbb{P}_S[B] &= \mathbb{P}_S\left[\mathbb{P}_{S_1,S_2 \text{ partition } S}\left[\bigcup_{h \oplus c \in \Pi^{\boldsymbol{\varepsilon}}_{H \oplus c}} B_{h \oplus c}|S\right]\right] \text{ by cond prob (like cond exp)} \\
&\leq \mathbb{P}_S\left[\sum_{h \oplus c \in \Pi^{\boldsymbol{\varepsilon}}_{H \oplus c}} \mathbb{P}_{S_1,S_2 \text{ partition } S}\left[B_{h \oplus c}|S\right]\right] \text{ by a union bound} \\
&\leq \left|\Pi^{\boldsymbol{\varepsilon}}_{H \oplus c}\right| \cdot 2^{-\varepsilon m/2} \leq \left(\frac{2\varepsilon m}{d}\right)^d 2^{-\varepsilon m/2} \text{ as } \Pi^{\boldsymbol{\varepsilon}}_{H \oplus c} \subseteq \Pi_{H \oplus c}, \text{ and prob doesn't dep on } S
\end{aligned}$$

So $\mathbb{P}[A] \leq 2\mathbb{P}[B] \leq \delta$ by above some calculus-y bounds. $\square$

Sample complexity LOWER bound

Theorem. *Given C is a concept class with $\mathrm{VCD}(C) \geq d \geq 25$, any PAC learning algorithm for C using $H \supseteq C$ requires at least $M := \max\left\{\frac{d-1}{32\varepsilon}, \frac{1}{4\varepsilon}\log\frac{1}{4\delta}\right\}$ examples*

Proof. For all L using fewer than M, there is a distribution $D, c \in C$ st $\mathbb{P}[\mathrm{err}(h_L) \geq \varepsilon] \geq \delta$. Suppose such an L exists, and so we can consider D as a samples S of size M.

1) $M \geq \frac{d-1}{32\varepsilon}$: let $T = \{x_1, ..., x_d\}$ be a set of size d shattered by C, and D be the distribution $D(x_1) = 1 - 8\varepsilon$, $D(x_j) = 8\varepsilon/(d-1)$ for $j > 1$.

We randomly choose a concept c, u.a.r over $\Pi_C(T)$.

Then run L with a sample $S \subseteq T$ of size $m = \frac{d-1}{32\varepsilon}$, labeled according to c.

We can assume that $h(x_1) = c(x_1)$ if $x_1 \in S$, as otherwise we could come up with L' that runs L and then corrects the output on x_1, since we have its label.

Let Z_i be the indicator for the event that the ith sample in S is from $x_2, ..., x_d$, so $\mathbb{P}[Z_i = 1] = 8\varepsilon$. Let $Z = \sum_i Z_i$, so $\mathbb{E}[Z] = 8m\varepsilon = \frac{d-1}{4}$

By a Chernoff bound, $\mathbb{P}[Z \geq \frac{d-1}{2} = 2\mathbb{E}[Z]] \leq \exp(-\frac{d-1}{12}) \leq \exp(-2)$

Let $\mathcal{E}$ be the event that $Z < \frac{d-1}{2}$, which implies that $x_1 \in S$, as this only occurs if $Z = m$. So $\mathbb{P}[\mathcal{E}] \geq 1 - \exp(-2) > 1/2$.

Conditioned on $\mathcal{E}$, $|S| = Z + 1 < \frac{d+1}{2}$ (as samples from $x_2, ..., x_d$ must repeat), so conditioned on $\mathcal{E}$, the dist. of concepts is uniform over a set size $2^{d-|S|} \geq 2^{(d-1)/2}$ (where obviously the seen examples are fixed, but the unseen ones could vary)

Then $\mathbb{P}_{x\sim D}[h(x) \neq c(x)|\mathcal{E}] = \sum_c \mathbb{P}[h(x) \neq c(x)|\mathcal{E}, "C" = c] \cdot \mathbb{P}["C" = c|\mathcal{E}] = \sum_c \cdot \frac{1}{2^{d-|S|}}$ $\square$

Consistent learner for linear threshold functions

$$\mathsf{LTF}_n := \{\mathbb{R}^n \ni \boldsymbol{x} \mapsto \mathbf{1}_{\geq 0}(\boldsymbol{w}\cdot\boldsymbol{x} + w_0) : \boldsymbol{w} \in \mathbb{R}^n, \|w\|_2 = 1, w_0 \in \mathbb{R}\}$$

(or other definitions of weight bounds).

An efficient consistent learner is essentially solving the LP:

$$w_0 + w \cdot \boldsymbol{x}_i \quad \geq \quad 0 \text{ for } i \text{ st } y_i = 1$$

4 Boosting

Weak learnability:

for a function $\gamma(\cdot,\cdot)$, L is a γ-weak PAC learner for C using H if $\forall n \geq 0, \forall c \in C_n$, for any D over X_n and $\delta \in (0, 1/2)$, given L has access to $\texttt{EX}(c, D)$, inputs $\text{size}(c)$,δ, γ outputs $h \in H_n$ st w.p. $\geq 1 - \delta$, $\text{err}(h; c, D) \leq \frac{1}{2} - \gamma(n, \text{size}(c))$.

L is efficient if H is polynomially evaluatable, $1/\gamma(n, \text{size}(c))$ is bounded in a poly in $n, \text{size}(c)$ and the running time of L is poly in $n, 1/\delta$ and $\text{size}(c)$.

AdaBoost algorithm:

inputs: training sample size m $((x_i, y_i))_{i=1}^m$, T= # of iterations, δ, the confidence parameter, and a weak-learning algorithm WL.

- set $D_1(m) = 1/m$
- for $t = 1...T$ do:
  - run WL on the sample, drawing examples according to D_t, with confidence parameter δ/T
  - recieve a hypothesis h_t.
  - $\varepsilon_t := \mathbb{P}_{(x,y)\sim D_t}[h_t(x) \neq y] = \sum_{i=1}^m \mathbf{1}(h_t(x) \neq y) D_t(i)$. note this is $\leq 1/2 - \gamma$ with probability $\geq 1 - \delta/T$
  - $\alpha_t := \frac{1}{2} \log\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)$
  - $D_{t+1}(i) := D_t(i) \cdot \exp(-\alpha_t y_i h_t(x_i))/Z_{t+1} = D_t(i)/Z_{t+1} \cdot \begin{cases} \exp \alpha_t & h_t(x_i) \neq y_i \\ \exp(-\alpha_t) & h_t(x_i) = y_i \end{cases}$
    - where the normalising constant $Z_{t+1} := \sum_{i=1}^m (D_t(i) \cdot \exp(-\alpha_t y_i h_t(x_i)))$

AdaBoost analysis:

After $T \geq \dfrac{\log 2m}{2\gamma^2}$ iterations, the training error of AdaBoost is 0 with probability $\geq 1 - \delta$.

For proof of said analysis, see notes/problem sheet

Usage: then use a ERM thing, like S1.

Lemma 4.3: VCD of THRESHOLDS$_k(H)$: (from Alex's class)

Given $\text{VCD}(H) = d$, we prove that $\text{VCD}(\text{THRESHOLDS}_k(H)) = O(kd \log kd)$

where $\text{THRESHOLDS}_k(H) := \left\{x \mapsto \text{sign}\left(\sum_{i=1}^k \alpha_k h_k(x)\right) : h_i \in H, \alpha_i \in \mathbb{R}, i \in [k]\right\}$

Difficult to prove by shattering sets

Consider the Growth function, and try estimating it: let $S = \{x_1, ..., x_m\}$

2 steps of prediction - to compute $\tilde{h}(x)$, first $h_i(x)$ for each i, and then apply the α_i's

so can separate S by separating points in $\mathbb{R}$. By applying all possible h's to S, we get at most $\Pi_H(m) \leq (\frac{me}{d})^d$ dichotomies.

let $S' = \left\{ \begin{array}{ccc} h_1(x_1) & \cdots & h_k(x_1) \\ \vdots & \ddots & \vdots \\ h_k(x_m) & \cdots & h_k(x_m) \end{array} \right\}$ is the set of points (=rows) in $\mathbb{R}^m$ we have when we fix a particular set of k $h_i \in H$.

Each column can be chosen up to $A = (\frac{me}{d})^d$ ways, so there are k times that possible "versions" of S' (on some fixed data set)

So then we pass this through a threshold function, which has VCD k (not $+1$ as no coefficient), so we apply Sauer's lemma again, so there are at most $B = (\frac{me}{k})^k$

So there are total $A \times B$ dichotomies, so

$\Pi_{\text{THRESHOLDS}_{k(H)}}(m) \leq (\frac{me}{d})^d (\frac{me}{k})^k$ (where some of these dichotomies will definitely be overcounted) $\leq m^{k(d+1)}$ for some bad bounding

So want to know when m is such that $2^m \leq \Pi_{\text{THRESHOLDS}_{k(H)}}(m)$, which gives us $m = O(kd \log(kd))$ - $m \log 2 \leq k(d+1) \log m$

5 Cryptographic hardness of learning

DCR problem

p, q two large primes, of the form $3k + 2$

$N := pq$

$\phi(n) := |\{k \in [n] : \gcd(k, n) = 1\}|$

here, $\phi(N) = (p-1)(q-1)$

note that 3 does not divide $\phi(N)$, as p, q are of the form $3k + 2$ (if so, then p or q is of the form $1 + 3k$)

$\mathbb{Z}_N^* := \{i : 0 < i < N, \gcd(i, N) = 1\}$, which is group under multiplication mod. N

$f_N : \mathbb{Z}_N^* \to \mathbb{Z}_N^*$ is defined by $f_N(y) \equiv y^3 \mod N$.

f_N is a bijection, as:

- $\gcd(3, \phi(N)) = 1$
- $\implies \exists d, d' \geq 1$ st $3d = \phi(N)d' + 1$ (Euclid's algorithm)
- $\implies (f_N(y))^d \equiv y^{3d} \equiv y^{\phi(N)d'+1} \equiv y \mod N$, as $y^{\phi(N)} \equiv 1 \mod N$ for all y

Statement of the problem:

Given N derived from p, q as above, input $x \in \mathbb{Z}_N^*$, output $y \in \mathbb{Z}_N^*$ st $y^3 \equiv x \mod N$

Statement of the assumption:

For any poly $P(\cdot)$, there does not exist an algorithm A st:

- given p, q are two $\boldsymbol{n}$**-bit** random primes of the form $3k+2$, $N := pq$,
- x is drawn randomly from $\mathbb{Z}_N^*$
- A runs in time $P(n)$ on inputs N, x and outputs $y \in \mathbb{Z}_N^*$ st $y^3 \equiv x \mod N$with probability $\geq 1/P(n)$.

The probability is over the draws of p, q, x and randomness in A.

DCRA in learning

suppose we were given a training sample $S = \{(x_i, y_i)\}_{i=1}^m$, where $y_i^3 \equiv x_i \mod N$, and the x_i are drawn randomly from $\mathbb{Z}_N^*$.

This sample is useless for learning, as we can generate arbitrary samples by picking y and cubing it, as f_N is a bijection so the distribution is the same.

Further, although $f_N^{-1}(x) \in \mathbb{Z}_N^*$, and thus isn't a boolean function, we can represent it as a bit string of length $2n$, so we have $2n$ boolean function, each returning a bit.

We can turn DCRA into a PAC problem as follows:

Initial problem:

input: training sample S

output: hypothesis $h : \mathbb{Z}_N^* \to \mathbb{Z}_N^*$ st $\mathbb{P}_{x \sim : \mathbb{Z}_N^*}[h(x) \neq f_n^{-1}(x)] \leq \varepsilon$.

DCRA says this problem has no efficient algorithm (by flipping from P to $1/P$ for ε in PAC definition)

We can further say that there is a PAC learning problem w/o an efficient algorithm by considering bit functions, as above - if all were PAC learnable then f_N^{-1} would be too.

Identifying a concept class

We now have a function that is hard to learn (as can demonstrate a specific distribution, minimum δ (given ε)), but want a class containing this function.

Result 1**:** $\exists$a poly $P(\cdot)$ st the class of concepts $C_n :=$ circuits of size $\leq P(n)$, $C = \bigcup C_n$, C is not PAC-learnable under DCRA.

Result 2: can reduce this to $C_n :=$circuits with depth bounded by $K \log n$ for a constant K, by cleverly computing powers, and giving some more inputs - see notes for details.

6 Exact Learning

Membership query oracle, $\texttt{MQ}(c)$: input $x \in X$, returns $c(x)$

Equivalence oracle, $\texttt{EQ}(c)$: input $h \in H$, returns equivalent if $h(x) = c(x)$ for all $x \in X$, otherwise returns a counterexample $x \in X$ st $h(x) \neq c(x)$

Exact Learning with MQ+EQ:

C is efficiently exactly learnable if $\exists$a polynomially-evaluatable hypothesis class H, a poly $p(\cdot, \cdot)$ and an algorithm L st:

- $\forall n \geq 1, \forall c \in C_n$, L, when given acces to $\texttt{MQ}(c), \texttt{EQ}(c)$ and $\text{size}(c)$ halts in time $p(n, \text{size}(c))$ and outputs a hypothesis $h \in H_n$ that is equivalent to c. All queries L makes to $\texttt{EQ}(c)$ must be of $h \in H_n$

Reducibility between learning problems, under PAC learning:

PAC learning C polynomially reduces to PAC learning $C\prime$ if:

- $\exists p$ poly and poly time computable func F st $\forall n F : X_n \to X'_{p(n)}$
- $\exists$ a map G st $\forall n : G(C_n) \subseteq C'_{p(n)}$ and $\forall x \in X_n, c \in C_n$, $c(x) = G(c)(F(x))$ (G doesn't have to be polynomially evaluatable - won't be computing G, just want to know it exists)

Theorem. *If C' is efficiently PAC learnable, and C reduces to C', then so is C*

Proof. let $c \in C$ be the target.

Generate $\texttt{EX}(c', D')$ by drawing $(x, y) \sim \texttt{EX}(c, D)$, and output $(F(x), y)$.

Run the alg for C' with this oracle to get $h' : X'_n \to \{0, 1\}$, and output a hypothesis for C, $h : X_n \to \{0, 1\}$ by $h(x) = h'(F(x))$

$\mathbb{P}_{x \sim D}[c(x) \neq h(x)] = \mathbb{P}_{x' \sim D'}[G(c)(F(x)) \neq h'(F(x))]$ □

Can't PAC learn acyclic DFAs:

C_1: class of circuits of depth $O(\log n)$- each node in the circuit can have multiple children - so a DAG, not a tree

C_2: formulae of size $\leq$ poly(n)

C_3: any function that can be represented by a log-space Turing machine (space on tape is $O(log$ input size$)$

C_4: ADFA

We have $C_1 \leq_{PAC} C_2 \leq_{PAC} C_3 \leq_{PAC} C_4$

	CONJ	3-T-DNF	Monotone DNF	DNF
Proper PAC	Y	N	N	N
PAC	Y	Y (via reductions to CONJ from 3-T-CNF)	?	<-=? (equally hard, se
PAC+MQ	Y	Y	Y lecs	= to above under cry

Noisy models

Learning with random classification noise:

- only the label y is flipped, and all are flipped with equal probability $\eta \in (0, 1/2)$
- new oracle $\texttt{EX}^\eta(c, D)$: draws $x \sim D$,returns $(x, c(x))$ with prob $1 - \eta$ and $(x, 1 - c(x))$ with prob η
- as $\eta \to 1/2$, learning is obviously harder, and we will find factors of $\frac{1}{1-2\eta}$ in all complexity bounds

PAC with RCN:

Let C be a concept class and H a polynomially-evaluatable hypothesis class. C is efficiently PAC-with-RCN learnable if there is an algorithm L st:

- $\forall n \geq 1; \forall c \in C_n; \forall D$ over $X_n; \forall \varepsilon \in (0, 1/2); \forall \delta \in (0, 1/2); \forall \eta \in (0, 1/2)$ if L, given access to $\mathbb{E}^\eta(c, D)$ and inputs $n, \varepsilon, \delta, \text{size}(c), \eta_0$ st $\eta \leq \eta_0 < 1/2$, outputs $h \in H_n$ st $\mathbb{P}[\text{err}(h; c, D) \leq \varepsilon] \geq 1 - \delta$ and runs in time poly in $n, \text{size}(c), \frac{1}{\varepsilon}, \frac{1}{\delta}, \frac{1}{1-2\eta_0}$

PAC-with-RCN for CONJUNCTIONS

let ℓ be a literal: $\mathbb{P}_{x\sim D}[\ell(x) = 0 \wedge c(x) = 1] =: p_\ell$ This probability will be zero if ℓ is in the target conjunction.

ℓ is significant if $\mathbb{P}[\ell(x) = 0] \geq \varepsilon/8n$

ℓ is harmful if $p_\ell \geq \varepsilon/8n$ (which implies significant too)

$h :=$ conjunctions of significant and not-harmful literals

$\mathbb{P}[h(x) \neq c(x)] = \mathbb{P}[h(x) = 1 \wedge c(x) = 0] + \mathbb{P}[h(x) = 0 \wedge c(x) = 1]$

- - first term is caused by insignificant literals, which should have been added (so h doesn't return 0 on those cases)
- and the second by significant & non-harmful literals (which can still be wrong with small probability)

$\leq 2 \times \varepsilon/8n \cdot 2n = \varepsilon/2$

Thus, by Chernoff bounds to estimate which literals are significant/harmful, we can generate a good approximation of h.

Statistical Query Learning:

Let C be a concept class, H a hypothesis class.

C is efficently SQ-learnable using H if $\exists$an algorithm L, polynomials p, q, r st

$\forall n \geq 1, \forall C \in C_n, \forall D$ over $X_n, \forall \varepsilon \in (0, 1/2)$, L with access to $\texttt{STAT}(c, D)$ and inputs $\varepsilon, \text{size}(c)$ does the following:

- $\forall$queries χ, τ: χ can be evaluated in time $q(n, \text{size}(c), 1/\varepsilon)$ and t is bounded by $r(n, \text{size}(c), 1/\varepsilon)$
- L halts in time $p(n, \text{size}(c), 1/\varepsilon)$ and outputs $h \in H_n$ st $\text{err}(h) \leq \varepsilon$

$\texttt{STAT}(c, D)$ is a new type of oracle:

- taking input $\chi : X \times \{0,1\} \to \{0,1\}$ and $\tau \in (0,1)$, the tolerance parameter
- returns $\hat{v} \in [v - \tau, v + \tau]$, where $v = \mathbb{E}_{x \sim D}[\chi(x, c(x)]$ - i.e. the expected value of the input function over the distribution

example use for conjunctions:

$\chi(x, y) = \begin{cases} 1 & \ell(x) = 0 \\ 0 & \text{otherwise} \end{cases}$ calculates for us $\mathbb{P}[\ell(x) = 0]$, which lets us decide whether a literal is significant

Theorem. *If C is efficiently SQ learnable, then C is efficiently PAC learnable*

Proof.

Overall idea: want to simulate STAT oracle just with the standard PAC oracle EX, which we do by simplifying the STAT oracle into 2 separate

Boolean trickery: use $\{-1, 1\}$ instead of $\{0, 1\}$ by the map $1 \to -1, 0 \to 1$, so $c : X \to \{-1, 1\}, \chi : X \times \{-1, 1\} \to \{-1, 1\}$

a query is:

$$\begin{aligned} \mathbb{E}_{c \sim D}[\chi(x, c(x)] &= \mathbb{E}[\chi(x, 1)\mathbf{1}(c(x) = 1)] + \mathbb{E}[\chi(x, -1)\mathbf{1}(c(x) = -1] \\ &= \mathbb{E}[\chi(x, 1)\frac{1 + c(x)}{2}] + \mathbb{E}[\chi(x, -1)\frac{1 - c(x)}{2}] \\ &= \frac{1}{2}\left(\underbrace{\mathbb{E}[\chi(x, 1)] + \mathbb{E}[\chi(x, -1)]}_{\text{target-independent queries}} + \underbrace{\mathbb{E}[\chi(x, 1) \cdot c(x)] - \mathbb{E}[\chi(x, -1) \cdot c(x)]}_{\text{correlational queries}}\right) \end{aligned}$$

New query model: for input $\phi : X \to \{-1, 1\}$ we can query $\mathbb{E}[\phi(x)]$ or $\mathbb{E}[\phi(x)c(x)]$, again with tolerances $\tau \in (0, 1)$

So an SQ-learning algorithm works prefectly well when built on top of these new queries (with tolerances appropriately divided on each split)

With just a noisy oracle, we can simulate a target-independent query with $O(\frac{1}{\tau^2} \log \frac{Q}{\delta})$ examples, for it to work with prob $\geq 1 - \delta$ to tolerance τ (where Q is the total number of queries, used for union bound)

We can also simulate a correlational query: note that the pair $(x, c(x) \cdot Z)$, where $\mathbb{P}(Z = -1) = \eta, \mathbb{P}(Z = 1) = 1 - \eta$ is equiv to $\texttt{EX}^\eta(c, D)$

So $\mathbb{E}_{(x,y) \sim \texttt{EX}^\eta(c,D)}[\phi(x)y] = \mathbb{E}_{x \sim D, Z \text{ indep}}[\phi(x)c(x)Z] = \mathbb{E}[\phi(x)c(x)] \cdot (1 - 2\eta)$, where the first term is the expectation of what we get from the noisy oracle, and the last term is what the query should return (times a constant...., so we need the tolerance on the estimate of the first term to be $\tau' := \tau(1 - 2\eta)$. Note the noise is independent, so this works. $\square$

If we don't know η: (n.b. we don't even need to know η_0, as $\leq 1/2$): let $\eta_i = i\Delta$ for $i = 1, 2, ..., \lceil \eta_0/\Delta \rceil$

run the algorithm with each η_i, to get hypotheses h_i. One of them is st $\exists i$ st $|\eta - \eta_i| \leq \Delta$, and so is "good"

So if we look at $\mathbb{E}_{(x,y)\sim \mathtt{EX}^\eta(c,D)}[h_i(x,y)] = \mathbb{E}_{(x,y)\sim \mathtt{EX}(c,D)}[h_i(x,y)(1-2\eta)]$, and choose the h_i with this expectation largest (so another Chernoff bound)

Aside on inclusions (specific to general)

1. SQ
2. PAC+RCN - R-PARITIES, by using alg below on silly class
3. PAC - PARITIES (def not in SQ, probably isn't in PAC+RCN)
4. PAC+MQ - DFA (not in PAC by DCRA)

Best known alg for parities with random classification noise is $2^{n/\log n}$

R-PARITY $= \{f_S : S \subseteq \{1...\log\log n \ \log n\}, |S| \leq \log n\}$ on $X_n = \{-1,1\}^n$

so |R-PARITY|= super poly(n) (as has logs)

But $R - PARITY$ isn't in SQ because the same proof we wrote for PARITIES is

this essentially makes the input larger without making the concepts larger

Non-SQ learnable problem:

Theorem. *PARITIES is not efficiently SQ-learnable: Precisely,*

if an algorithm L makes q queries with tolerance $\geq \tau_0$ st $\frac{q}{\tau_0} < 2^n - 2$ then the algorithm cannot learn PARITIES.

Proof. Again, let $X_n = \{-1,1\}^n$, so parity functions are of the form $f_S(x) = \prod_{i\in S} x_i$ for $S \subseteq [n]$.

Let D be uniform on X_n, and fix

Note $\mathbb{E}_{x\sim U}[f_S(x)] = \begin{cases} 0 & S \neq \emptyset \\ 1 & S = \emptyset \end{cases}$

If we have f_S, f_T then $\mathbb{E}_{x\sim U}[f_S(x)f_T(x)] = \mathbb{E}_{x\sim U}[f_{S\Delta T}(x)] = \begin{cases} 0 & S \neq T \\ 1 & S = T \end{cases}$, where $S\Delta T$ is the symmetric difference.

consider the 2^n-dimensional vector space $V = \{f : X_n \to \mathbb{R}\}$ - by above, the parity functions form an orthonormal basis of V with the inner product $\langle g, g' \rangle := \mathbb{E}_{x\sim U}[g(x)g'(x)]$.

So for $g \in V$, write $g(x) = \sum_{S\subseteq[n]} g(S), f_S(x)$, where $g(S) := \langle g(x), f_S(x) \rangle$ (slight notational sillliness.

By Parseval's identity, $\sum_S g(S)^2 = \mathbb{E}_{x\sim U}[g(x)^2]$

Let us assume the algorithm is deterministic, and only makes correlational queries:

- this is justifiable, as we "allow" the algorithm to know the distribution is uniform, so the target-independent queries are pointless.

Given the queries made are $\chi_1, ..., \chi_q : X_n \to \{-1, 1\}$, we decide to return 0 to all of them.

How many S are there, st $|\mathbb{E}[\chi_1(x) f_S(x)]| \geq \tau_0$?

$\chi_1(x) = \sum_S \chi_1(S) f_S(x)$, $\chi_1(S) = \mathbb{E}[\chi_1(x) f_S(x)]$. Since $\mathbb{E}[\chi_1^2(x)] = 1 = \sum \chi_1(S)$, there are at most $1/\tau_0^2$ S st $|\chi_1(S)| \geq \tau_0$.

After q such queries, we have ruled out $\frac{q}{\tau_0^2}$ possible target functions, so by the limit on q we have two possible parity functions consistent with the query answers we have given.

So if S_1, S_2 are the two subsets consistent with the answers we have given, so

$$\mathbb{P}[h(x) \neq f_{S_1}(x)] + \mathbb{P}[h(x) \neq f_{S_2}(x)] \geq \mathbb{P}[f_{S_1}(x) \neq f_{S_2}(x)] = 1/2$$

Thus one of these properties is $\geq 1/4$, so take $\varepsilon = 1/5$ in the original definition. □

If we have a random queries instead, there is a small factor in # of queries.

Occam's razor for PAC+RCN: need to ignore efficiency

consistent learner is just least bad....

Learning real-valued Functions

Instance space is X_n, e.g. $\mathbb{R}^n$, have a class of functions $\mathcal{F}$ of $f : X_n \to \mathbb{R}$

target function $f \in \mathcal{F}$

Loss function is $\ell : \mathbb{R} \times \mathbb{R} \to \mathbb{R}_{\geq 0}$

have some distribution D over $X_n \times \mathbb{R}$, let μ be the marginal distribution of D over X_n

the risk of $g : X_n \to \mathbb{R}$ is $R(g) := \mathbb{E}_{(x,y)\sim D}[\ell(g(x), y)]$

we can require that the distribution D perfectly matches the target - i.e. supp$(D) \subseteq \{(x, f(x) : x \in X_n\}$ which is the analogue of PAC

or we can assume that there is some noise in D: the **zero-mean noise model**

$\mathbb{E}[y|X] = f(X)$ - so noise is centred around the target function

So it is reasonable to ask for g st $\mathbb{E}_{x\sim\mu}[(g(x) - f(x))^2] \leq \varepsilon$.

A more general setting (non-realizable/agnostic) is to find g st $R(g) - \inf_{f\in F} R(f) \leq \varepsilon$

Given a sample $S \sim D^m$, $S = \{(x_1, y_1), ..., (x_m, y_m)\}$ the empirical risk is $\hat{R}_S(g) = \frac{1}{m}\sum_{i=1}^m \ell(g(x_i), y_i)$

$$\begin{aligned}\mathbb{E}[(g(x)-y)^2] &= \mathbb{E}_x[(g(x)-f(x))^2] + \mathbb{E}_{x,y}[(f(x)-y)^2] + 2\mathbb{E}_{x,y}[(g(x)-f(x))(f(x)-y)] \\ &= \mathbb{E}[(g(x)-f(x))^2] + \mathbb{E}[(f(x)-y)^2]\end{aligned}$$

where the last term disappears because $f(x) - y$ involves an expectation over y, which is 0

linear regression:

$\mathcal{F} : \{x \mapsto w \cdot x : w \in \mathbb{R}^n\}$

closed form solution to ERM is $\hat{w} = (X^T X)^{-1} X^T y$

goal: find $\hat{w}$ st $\mathbb{E}[(\hat{w} \cdot x - w^* \cdot x)^2] \leq \varepsilon$, where w^* is the target weights.

we can also consider $\|\hat{w} - w^*\|_2^2$, but this is a different thing to minimise (estimation error)

convex optimisation:

convex set $K \subseteq \mathbb{R}^n$, convex function f

diam$(K) = \sup_{w,w' \in K} \|w - w'\|_K$

B st diam$(K) \leq B$

Lipschitz bound L

if differentiable as well, then $f(x) \geq f(x_0) + \nabla f(x_0) \cdot (x - x_0)$

projected gradient descent

$w_0$in K

for $t = 0...T-1$

$w'_{t+1} = w_t - \eta \nabla f(w_t)$

$w_{t+1} = \Pi_K(w'_{t+1})$ project back into K

output average $1/T \sum_1^T w_t$

non-expansion lemma thing

for T steps with $\eta = \frac{\beta}{L}\sqrt{1/T}$ satisfies $f(\frac{1}{T}\sum_{t=1}^T w_t) \leq \min_{w \in K} f(w) + \frac{BL}{\sqrt{T}}$

Lecture 14/11/22

let $\ell(\hat{y}, y) = (\hat{y} - y)^2$, so squared error

$R(g) = \mathbb{E}_{x,y \sim D}[\ell(g(x), y))]$ is the error

also have standard empirical error

given $\mathcal{F}$ is a class of functions

realizable setting, so $\exists f^* \in \mathcal{F}$ st $\mathbb{E}[y|x] = f^*(x)$

$\mathcal{G}$ is a class of functions st $\mathcal{F} \subseteq \mathcal{G}$

algorithm: find $g \in \mathcal{G}$ st $\hat{R}_S(g) \leq \min_{g \in G} \hat{R}_S(g) + \varepsilon$ - this is solvable with optimisation, for a fixed sample

other half of the question: w.p. $\geq 1 - \delta$, for all $g \in \mathcal{G}$ $\|\hat{R}_S(g) - R(g)\| \leq \varepsilon$

so together, we can find a function with suitably low actual rist

re-covering some AFoL:

in this realizable setting, $\mathbb{E}[\ell(\hat{g}(x), f^*(x))] = R(\hat{g}) - R(f^*)$

Further, this is equal to $\Big(R(\hat{g}) - \hat{R}_S(\hat{g})\Big) + \Big(\hat{R}_S(\hat{g}) - \hat{R}_S(f^*)\Big) + \Big(\hat{R}_S(f^*) - R(f^*)\Big)$, and with the results above we can bound this by 3ε.

in the linear setting:

w^* defines f^*, and $\|w^*\| \leq W$, so $\mathbb{E}[y|x] = f^*(x) = w^* \cdot x$

Define B, M st $\forall (x, y) \in \text{support}(D), \|x\| \leq B, \|y\| \leq M$

Then let $g(w) := \hat{R}_S(w) = \frac{1}{m} \sum_{i=1}^m (w \cdot x_i - y_i)^2$, and consider $K = \{w \in \mathbb{R}^n : \|w\| \leq W\}$.

g is a convex, differentiable function, st $\nabla g(w) = \frac{2}{m} \sum_{i=1}^m (w \cdot x_i - y_i) \cdot x_i$. so $\|\nabla g(w)\| \leq 10(BW + M)B$.

Generalized linear models

$g : \mathbb{R}^n \to \mathbb{R}$

$g(x) = u(w \cdot x)$, where $w \in \mathbb{R}^n, \|w\| \leq W, u : \mathbb{R} \to \mathbb{R}$ is a monotonically-increasing Lipschitz function.

We consider u fixed for a particular class, so the class $\mathcal{F}$varies only over w.

For example, we could take the sigmoid function $u(z) = \frac{1}{1+\exp(-z)}$

The function $h(w) = \min_w \frac{1}{m} \sum_{i=1}^m (u(w \cdot x_i) - y_i)^2$ is no longer convex.

To fix this, we try and come up with a convex function to relate to this.

If $h'(w) = (u(w \cdot x) - y)^2$, then $\Delta h'(w) = 2(u(w \cdot x) - y)u'(w \cdot x) \cdot x$

This is messy

surrogate loss:

Now consider $\ell(w) := \int_0^{w \cdot x} u(z) - y \, dz$, so $\nabla \ell(w) = (u(w \cdot x) - y)x$

so consider $\mathbb{E}_y[\ell(w; x, y)] = \mathbb{E}_y[\int_0^{w \cdot x} u(z) - y \;\; dz] = \int_0^{w \cdot x} u(z) - \mathbb{E}[y] \;\; dz = \int_0^{w \cdot x} u(z) - u(w^* \cdot x) \; dz$

$\mathbb{E}_y[\ell(w; x, y)] - \mathbb{E}_y[\ell(w^*; x, y)] = \int_{w^* \cdot x}^{w \cdot x} u(z) - u(w^* \cdot x) \; dz$ (if $w^* \cdot x \leq w \cdot x$, otherwise the other way round)

- argument for following bound:
- function depends on z only, is 1-Lipschitz, and monotonically increasing.
- it exists in a range $[0, u(w \cdot x) - u(w^* \cdot x)]$, and must hit this final value

- thus, since it is $1-L$ it's slope must be 1, and must stay above a right triangle of side $(u(w \cdot x) - u(w^* \cdot x))$ for both.

$\geq \frac{1}{2}(u(w \cdot x) - u(w^* \cdot x))^2$ by Lipschitz-ness THINK ABOUT THIS STEP!!!

so w^* is a minimiser of $\mathbb{E}_y[\ell(w; x, y)]$

surrogate risk: $R^\ell(w) := \mathbb{E}_{x,y}[\ell(w; x, y)]$ - w^* is a minimizer of this, so if we find w st $R^\ell(w) \leq R^\ell(w^*) + \varepsilon$, then we have that $\mathbb{E}[\frac{1}{2}(u(w \cdot x) - u(w^* \cdot x))^2] \leq \varepsilon$, so we actually get a bound on the (real) squared loss risk.

Note that the Hessian of ℓ is $u'(w \cdot x)(xx^\top)$, which is positive semi-definite, as $u' \geq 0$ as u increasing.

Convex losses for classification:

Either linear models $f(x) : w \cdot x$, using $\hat{y} := \text{sign}(f)$ as our actual classifier. Note that a u is thus completely pointless here

suaared loss: $\mathbf{1}(\hat{y} \neq y) \leq (f(x) - y)^2$

logistic loss: $\ln(1 + \exp(-y(f))$: $\mathbf{1}(\hat{y} \neq y) \leq \ln(1 + \exp(-y(f))/\ln(2)$

absolute loss: $\mathbf{1}(\hat{y} \neq y) \leq |f(x) - y|$

exponential: $\mathbf{1}(\hat{y} \neq y) \leq \exp(-yf(x))$

hinge loss: $\mathbf{1}(\hat{y} \neq y) \leq \frac{1}{\gamma}\max\{0, -yf(x) + \gamma\}$

So if we can minimize these RHSes, we know that we are doing well on classification.

(Empirical) Rademacher complexity

note lack of monotone u in prev lecture means we no longer have efficiency, as we can learn parities with noise.

Let $\mathcal{G}$ be some class of functions $f : X \to [a, b]$

D is a distribution over X

$S = (z_1, .., z_m) \sim D^m$ an iid sample

$\phi(S) = \phi(z_1, ..., z_m) = \sup_{g \in \mathcal{G}} \left\{\mathbb{E}_{z \sim D}[g(z)] - \frac{1}{m}\sum_{i=1}^m g(z_i)\right\}$ - i.e. the difference over all g between the true expectation and an empirical estimate

notation: let $\hat{\mathbb{E}}_S[g] = \frac{1}{m}\sum_{i=1}^m g(z_i)$ be the empirical expectation over the sample $S = \{z_1, ..., z_m\}$

$$\begin{aligned}
\mathbb{E}_S\phi(S) &= \mathbb{E}_S[\sup_g \left\{\mathbb{E}g - \hat{\mathbb{E}}_S g\right\}] \\
&= \mathbb{E}_S[\sup_g(\mathbb{E}_{S'}(\hat{\mathbb{E}}_{S'}g - \hat{\mathbb{E}}_S g))] \\
&= \mathbb{E}_S\mathbb{E}_{S'}\sup_g(\hat{\mathbb{E}}_{S'}g - \hat{\mathbb{E}}_S g)] \\
&= \mathbb{E}_{S,S'}\sup_g \frac{1}{m}\sum_{i=1}^m g(z_i') - g(z_i) \\
&= \mathbb{E}_{\sigma,S,S'}\sup_g \frac{1}{m}\sum_{i=1}^m \sigma_i(g(z_i') - g(z_i)) \\
&\leq 2\mathbb{E}_{\sigma,S}\sup_g \frac{1}{m}\sum_{i=1}^m \sigma_i g(z_i) \text{ by splitting sup on 2 terms}
\end{aligned}$$

where σ is a length m vector of iid $-1, 1$ Rademacher RVs σ_i st $\mathbb{P}[\sigma_i = 1] = \mathbb{P}[\sigma_i = -1] = 1/2$.

We can specialise to csondiering rish

Given $\mathcal{G}$ be some class of functions $f : X \to [a, b]$

$S = (z_1, .., z_m)$ (a multiset) , the **empirical Rademacher complexity is** $\widehat{\mathrm{RAD}}_S(\mathcal{G}) := \mathbb{E}_\sigma \sup_g \frac{1}{m}\sum_{i=1}^m \sigma_i g(z_i)$

so if D is a distribution over X, the **Rademacher complexity is** $\mathrm{RAD}(\mathcal{G}) := \mathbb{E}_S[\widehat{\mathrm{RAD}}_S(\mathcal{G})] = \mathbb{E}_{\sigma,S}\sup_g \frac{1}{m}\sum_{i=1}^m \sigma_i g(z_i)$

so if S is shattered by $\mathcal{G} \subseteq \{g : X \to \{0, 1\}\}$, then $\widehat{\mathrm{RAD}}_S(\mathcal{G}) = 1$

If $\mathcal{G}$ contains the constant $+1, -1$ functions, then $\widehat{\mathrm{RAD}}_S(\mathcal{G}) \geq \Omega(\frac{1}{\sqrt{n}})$

McDiarmid's inequality: X is a set, $f : X^m \to \mathbb{R}$ st $\forall i\ \exists c_i$ st $\forall z_1, ..., z_m, z_i' \in X$ $|f(z_1, ..., z_i, ..., z_m) - f(z_1, ..., z_i', ..., z_m)| \leq c_i$

$Z_1, , ..., Z_m$ are iid RVs taking values in X

then $\forall \varepsilon > 0\ \mathbb{P}[f(Z_1, ..., Z_m) \geq \mathbb{E}[f(Z_1, ..., Z_m) + \varepsilon] \leq \exp(\frac{-2\varepsilon^2}{\sum_i c_i^2})$

We wish to apply this to ϕ from earlier, which is true with $c_i \leq \frac{2(b-a)}{m}$ as the g_i are bounded

bounding $\phi(S)$ by rad + $\sqrt{\frac{l \log 1/delta}{2m}}$

Compositition results:

$\mathcal{F}_1 + \mathcal{F}_2 := \{f_1 + f_2 : f_1 \in \mathcal{F}_1, f_2 \in \mathcal{F}_2\}$

$\mathrm{RAD}_m(\mathcal{F}_1 + \mathcal{F}_2) \leq \mathrm{RAD}_m(\mathcal{F}_1) + \mathrm{RAD}_m(\mathcal{F}_2)$

$\mathrm{RAD}_m(c \cdot \mathcal{F}) = |c| \mathrm{RAD}_m(\mathcal{F})$

Talagrand's lemma:

$S \subseteq X, |S| = m$, $\mathcal{F}$ a class of functions

$\phi_1, ..., \phi_m$ are $L-$Lipschitz, then

$\mathbb{E}_\sigma[\sup_{f\in\mathcal{F}} \frac{1}{m}\sum_{i=1}^m \phi_i(f(z_i))] \leq L \cdot \widehat{\mathrm{RAD}}_S(\mathcal{F})$

Examples:

$\mathcal{F} = \{x \mapsto w \cdot x : \|w\| \leq W\}$, $\|x\| \leq B, \|y\| \leq M$ for $x, y \in$ supp(D)

Given $S = \{(x_1, y_1), ..., m\}$

$$\begin{aligned}
\widehat{\mathrm{RAD}}_S(\mathcal{F}) &= \mathbb{E}_\sigma \sup_w \frac{1}{m}\sum_{i=1}(w \cdot x_i)\sigma_i \\
&= \mathbb{E}_\sigma \sup_w w \cdot \frac{1}{m}\sum_{i=1} x_i\sigma_i \\
&\leq \mathbb{E}_\sigma \sup_w W \cdot \|\frac{1}{m}\sum_{i=1} x_i\sigma_i\| \\
&= W \cdot \mathbb{E}_\sigma\|\frac{1}{m}\sum_{i=1} x_i\sigma_i\| \\
&\leq W\sqrt{\frac{1}{m^2}\sum_i \|x_i\|^2} \text{ CAUTION, HARD!!!} \\
&\leq WB/\sqrt{m}
\end{aligned}$$

now consider $\mathcal{G}_1 = \{(x, y) \mapsto w \cdot x - y : \|w\| \leq W\}$

$g \in \mathcal{G}_1$, so $g_1(x, y) \in \{-BW - M, BW + M\}$

$\mathcal{G}_2 = \{(x, y) \mapsto \phi(g(x, y) : g \in \mathcal{G}_1\}$

where $\phi(z) = z^2$ for squared loss, so $\phi'(z) = 2z$ and is $2(BW + M)$ on $[-BW - M, BW + M]$

so $\widehat{\mathrm{RAD}}_S(G_2) \leq (2BW + M)\frac{BW}{\sqrt{m}}$

so for $f \in \mathcal{F}$,we have that $|\mathbb{E}[(f(x) - y)^2] - \hat{\mathbb{E}}[(f(x) - y)^2]| \leq \frac{2(BW+M)(BW)}{\sqrt{m}}$

Online learning

Online learning/Mistake-bounded model

constantly get data, may not be from the same dist

instance space X

Learning algorithm

At time t, the algorithm gets data point $x_t \in X$, and must predict $\hat{y}_t \in \{0, 1\}$, and **after that** gets $y_t \in \{0, 1\}$.

So up to time t, the mistakes of the algorithm = MISTAKES$(T) = \sum_{s=1}^t \mathbf{1}(\hat{y}_t \neq y_t)$

have a concept class C, guarantee that $\exists c \in C$ st $\forall t$ $y_t = c(x_t)$ (otherwise learning isn't really possible, not without some other similar (maybe weaker) guarantee)

we want MISTAKES to remain small and finite even as $t \to \infty$ (note that 2^n is often trivial by memorising data, as gets it wrong on each example at most once)

for now, we allow the algorithm to remember all previous $x_t, y_t, (\hat{y}_t)$

Online learning Conjunctions

starting assumption: add everything

remove term(s) that disagree after each example - note this doesn't require storing any data ecept the hypothesis (and I suppose the last one)

Mistake bound: $\leq n + 1$

At the start, h_1 has $2n$ literals. When the first mistake occurs, exactly n literals are removed (as must remove 1 of each pair $x, \neg x$): for each later mistake at least one literal is dropped. Thus, the number of mistakes cannot exceed n + 1 (as that would require dropping more than $2n$ literals).

Mistake-bounded learning:

a concept class C is learnable with a mistake bound M if there exists a learning algorithm L st that $\forall c$ on any input sequence $(x_t, y_t)_{t=1}^{\infty}$ st $y_t = c(x_t)$ for all t, that $\forall t$ MISTAKES$(t) \leq M$

we can consider it as running $(\hat{y}_{t+1}, S_{t+1}) = f(S_t, x_{t+1}, y_t)$

and at time t we allow running time of f to be poly in $n, t, \text{size}(c)$ (where n is the size of the instance space)

space-based learning algorithms: at time t, it can store a "sketch" S_t st $|S_t| \leq$ poly$(n, \text{size}(c))$ [or alternatively $|S_t| \leq$ poly$(n, \text{size}(c), \log t)$ if we allow it to know what time it is]

efficiency: we require that the function f is efficiently computable

comparison to `EQ`

so if C can be learnt with mistake bound M, then C can be learnt with `EQ` in $M+1$ queries: at each step, we query `EQ` with the current hypothesis - if equivalent, we are done, otherwise we pass the counterexample to it (= a mistake), so must be at most $M + 1$ steps, as it makes a mistake on each step.

note this does require extending `EQ` to any representation of the function.

if C can be learnt exactly with $\leq q$ `EQ` queries, then it can be learnt with mistake bound q:

let A be the `EQ`alforithm. Let h_i be the ith query to `EQ`.

We use h_i to answer online learning queries, until it makes a mistake, and pass that back to A as a counter-example (thus when we get an h_i that happens to be consistent with c, though we will never know for certain, A is just left hanging forever).

This must happen at most q times, as that is the bound on the # of queries.

comparison to PAC: if C is (efficiently) online learnable with MB $M \leq$ poly$(n, \text{size}(c))$ then C is (efficiently) PAC learnable

do this via EQ:

so if A learns C with $\leq q$ EQ queries:

let h_1 be the first query made by A. If $\mathrm{err}(h_1) \geq \varepsilon$, then in $m =$???? examples we will??? find a mistake, which we pass back to A to get h_{i+1}

$\mathbb{P}[h_1$ makes no mistakes after m draws$] \leq (1-\varepsilon)^m \leq \exp(-\varepsilon m)$

if we go m steps without making a mistake, return h_i, or if we end up with M h's, then return the last as it will be correct.

total time is $\leq m, Bm$

Halving algorithm:

- let $C_1 = C$
- for $t = 1, 2.....$
  - given x_t, compute $c(x_t)$ for all $c \in C_t$
  - set $\hat{y}_t$ to be the majority label
  - observe y_t, and let $C_{t+1} = \{c \in C_t : c(x_t) = y_t\}$

the mistake bound of this is $\leq \log |C|$ (alg only really makes sense in the first place if C is finite)

relation to VC dim:

claim: $\mathrm{VCD}(C) \leq \max_t$ MISTAKES(t) for any mistake-bounded algorithm for C

let S be some shattered set, and then MISTAKES $\geq |S|$, as keep having concepts consistent with unseen points in S until we have seen all of S

some examples:

dictator functions: $c_i(x) = x_i$ for each $i \in [n]$

algorithm for this on specific instance space $X =$ set of basis vectors $1000, 0100, 0010, 0001$

always output 0, mistake bound $= 1$, as when we get a mistake, we know exactly which function was expected

on other instance spaces, get $\sim \log n$ again - prove?

binary search concept idea:

$X = \{1, ...2^n\}$, $c_i(x) := 0$ if $x < i$, otherwise 1, $C = \{c_i : i \in 1...2^n\}$. $\mathrm{VCD}(C) = 1$, but the mistake bound is n by a binary-search style argument

Online learning algorithms

Perceptron algorithm

$f : \mathbb{R}^n \to \{-1, 1\}$, concept class is half-spaces through the origin, so

$f_w(x) = 1$ if $w \cdot x \geq 0$, otherwise 0

The algorithm:

- Set $w_0 = 0 \in \mathbb{R}^n$
- for $t = 1, 2, ...$
  - recieve $x_t \in \mathbb{R}^n$
  - predict $\hat{y}_t :=$ sign$(w_t \cdot x_t)$
  - recieve y_t
  - if $\hat{y}_t \neq y_t$, set $w_{t+1} := w_t + y_t x_t$, otherwise no change.

Analysis/interpretation

let w^* be the true weights.

For any given weights w, these predict $+$ on points in the same half-space as them, so the mistake between w^* and w_t,say, is the symmetric difference of the two regions.

The update step $w_{t+1} := w_t + y_t x_t$ "rotates" towards the true weights

Theorem. *suppose $\|x_t\| \leq D$ for all t, let w^* be the true separator, st $y_t =$ sign$(w^* \cdot x_t)$, $\|w^*\| = 1$ and $|w^* \cdot x_t| \geq \gamma$ for all t. Then MISTAKES of the perceptron algorithm is $O(D^2/\gamma^2)$*

Proof. (using lemma below)

So $\gamma m_t \leq w_t \cdot w^* \leq \|w_t\|\|w^*\| \leq \sqrt{m_t} D_t$, so $m_t \leq D^2/\gamma^2$ (where we use the lemma below, and the Cauchy-Schwarz inequality. □

Lemma. *Let m_t be the # of mistakes made up to time t. Then $w_t \cdot w^* \geq m_t \gamma$, and $\|w_t\|^2 \leq m_t D^2$*

Proof. So $w_{t+1} = w_t + y_t x_t$.

$w_{t+1} \cdot w^* = w_t \cdot w^* + y_t x_t \cdot w^* \geq \gamma m_t + y_t x_t \cdot w^* \geq \gamma(m_t + 1)$, using induction in the first term, and for the second, the margin.

Thus we induct on the steps mistakes are made on.

Similarly,

$\|w_{t+1}\|^2 = \|w_t\|^2 + \|y_t x_t\|^2 + 2 \underbrace{y_t w_t \cdot x_t}_{<0} \leq \|w_t\|^2 + D^2 \leq m_{t+1} D^2$, where the < 0 bound is as this is a mistake, and $\|y_t x_t\| \leq D$ as $y_t \in \{-1, 1\}$ □

Note with a fair no. of data points, we could just run the LP consistent learner for offline learning, as $\mathrm{VCD} = n$ or $n + 1$

Monotone disjunctions

concept class: $f_S(x) = 1$ if at least one of a set $S \subseteq \{x_1, ..., x_n\}$ is satisfied, where $|S| \leq k$

could write this as a linear threshold function - e.q. sign$(\sum_{i \in S} x_i - 1/2)$ (note $1/2$, can put that into an extra dimension)

here, $D = \sqrt{n}$, and $\gamma = c/\sqrt{k}$ for a constant c, so the mistake bound is $O(nk)$

the VCD $\leq O(k \log n)$ by counting no. of subsets of size k

or a consistent learner:

- add all to S
- whenever 0, remove all vars which are 1.

Winnow algorithm for monotone disjuctions

[Littlestone '88]

- Start with $w^1 = \{1, ..., 1\}$, which will stay integers at all times
- for $t = 1, ...$
  - recieve $x_t \in \{0, 1\}^n$
  - $\hat{y}_t = 1(w^t \cdot x_t \geq n/2)$
  - recieve y_t, and if different:
    - if $\hat{y}_t = 1, y_t = 0$
      - set $w_i^{t+1} = 0$ for all i st $(x_t)_i = 1$
    - if $\hat{y}_t = 0, y_t = 1$
      - set $w_i^{t+1} = 2w_i^t$ for all i st $(x_t)_i = 1$

Explanation:

- first mistake type: **elimination steps (E)** - these terms are definitely not in the target S
- second type: **promotion steps (P)** we had the correct terms already, but didn't prioritise them.

Winnow algorithm analysis

- claim 1: $\forall t, \forall i \; w_i^t \leq n$:
  - any weight that may increase in a P step must be $< n/2$, otherwise it would have successfully predicted $\hat{y}_t$
- claim 2: $\forall t$, $\#P \leq k \log n$

  - in every promotion step, at least one "relevant" variable has its weight doubled
  - there are $\leq k$ relevant variables - the ones in the taget
  - each variable can be doubled $\leq \log n$ times
- claim 3: $\#E \leq \#P + 2$
  - let $T_t := \sum_{i=0}^{n} w_i^t$ be the total weight, which must obviously be non-negative at all t
  - at an elimination step, T_t decreases by $\geq n/2$
  - at a promotion step, T_t increases by $< n/2$
  - Initially, $T_1 = n$, so $0 \leq T_t \leq n + P(n/2) - E(n/2)$

So the $\#$ of mistakes is bounded by $2k \log n + 2$

This is in a sense equivalent to have a margin condition, because we only use integer weights!

Online learning with Expert advice

Problem:

- multi-round game between a decision maker, and an environment (which may be adversarial)
- we have a finite time horizon T, where the game ends - i think....
- the decision maker has n "options"/ "experts" A_i
- at each round, the decision maker must pick a distribution over these options, i.e. $x_t \in \Delta_n = \{x \in \mathbb{R}^n : \sum_i x_i = 1\}$
- then the environment picks a loss vector $\ell_t \in [0,1]^n$
- therefore the loss at time t is $\ell_t \cdot x_t$
- so the total loss is $\mathrm{Loss}(DM) = \sum_{t=1}^{T} x_t \cdot \ell_t$

adaptive adversary: the environment has access to previous choices of x_t and the current one too

oblivious adversary: the environment has no access to choices of x_t (i.e might as well choose losses at the start)

Best option in hindsight

Choose an option according to how it performed on previous rounds.

$\text{Loss}(A_i) = \sum_{i=1}^{t} \ell_{t,i}$

$\text{Regret}(DM) = \text{Loss}(DM) - \min_{x^* \in \Delta_n} \sum_{t=1}^{T} \ell_t \cdot x^*$ captures the idea of the environment being bad, in the sense that it compares the decisions made to the best possible option given the adversarialness of the environment. Note the strategy we are comparing to is a fixed strategy - there is no t subscript on x^*

note the regret may be negative (as we are allowed to change strategies, unlike the comparison strategy)

hedging idea: spread prob. mass around the actions, so the (adverserial) environment can't make the loss bad just on what we picked, so the regret doesn't change (as much)

Strategy 1: "follow the leader"

- pick x_t to be the indicator/one-hot of the best historical action
- this doesn't work at minimising regret:
  - on just two actions, A, B, first set $\text{Loss}(A) = 1/2, \text{Loss}(B) = 0$, then alternate setting them $1, 0$ and $0, 1$ forever
  - note the $1/2$ is important, to ensure the previous losses add up correctly
  - so $\text{Loss}(FTL) = T\ (-1)$, and $\text{Regret}(FTL) = T/2$, as the best action is choosing B in all cases

Multiplicative weight update algorithm

$w_1 = (1, ..., 1) \in \mathbb{R}^n$

for $t = 1, ..., T$:

- let $x_t = w_t / Z_t$, where $Z_t := \sum_{i=1} w_{t,i}$
- observe the loss vector ℓ_t (and suffer loss $x_t \cdot \ell_t$)
- $w_{t+1,i} := w_{t,i} \cdot \exp(-\eta \ell_{t,i})$

This is rather like a softmin

we (might) prove that $\text{Regret}(MWUA) \leq 2\sqrt{T \log n}$, given $\eta = \sqrt{\frac{\log n}{T}}$

this is tight: for example, on 2 actions the same alternating thing (randomly assigning 1,0 loss o actions), the expected loss of some??? DM is $T/2$, and the expected loss of the best constant strategy is $T/2 - c\sqrt{T}$

Boosting

$(x_1, y_1), ..., (x_m, y_m)$, hypothesis class H (for weak learning)

DM over m actions

- DM picks d_t over $\{1, ..., m\}$
- the environment does weak learning by picking h_t st $\text{err}(h_t; d_t) \leq 1/2 - \gamma$, and lets $\ell_{t,i} = \mathbf{1}(h_t(x_t) - y_t)$

By doing this, the MWUA algorithm gives us exactly the AdaBoost algorithm.

so we have that $d_t \cdot \ell_t \geq \frac{1}{2} + \gamma$

$\text{Loss}(DM) \geq \frac{T}{2} + \gamma T$

$\text{Loss}(i) = \sum_{i=1}^{T} \mathbf{1}(h_t(x_i) = y_i)$

so $\text{Regret}(DM) \leq 2\sqrt{T \log n}$

and so $\text{Loss}(i) \geq T/2 + \gamma T - 2\sqrt{T \log m} > T/2$ for $T \geq \log n)/\gamma^2$

so majorities is correct on all m examples.

Zero sum games

So we have two players, A, B, where A has actions $1...n$, B has $1..m$. There is a matrix $P_{i,j}$ for $i \in [n], j \in [m]$ which is the payoff to player A if A chooses action i and B chooses j, and $P_{i,j} \in [0, 1]$

Von Neumann's Min-Max theorem

if B can see A's action, then A's optimal strategy is to choose $\max_i \min_j P_{i,j}$, and if B goes first, then A will pick $\min_j \max_i P_{i,j}$

we have that $\max_i \min_j P_{i,j} \leq \min_j \max_i P_{i,j}$ **not convinced I have setup correct!!!!!!!!!!!!!!!**

Distributions over actions

So A, B each choose distributions over the actions $\sigma_A, \sigma_B \in \Delta_n, \Delta_m$ respectively

aagain

$\max_{\sigma_A} \min_{\sigma_B} P_{\sigma_A, \sigma_B} \leq \min_{\sigma_A} \max_{\sigma_B} P_{\sigma_A, \sigma_B}$ where $P_{\sigma_A, \sigma_B} = \mathbb{E}_{i \sim \sigma_A, j \sim \sigma_B}[P_{i,j}]$, but these are actually equial.

Player 1 uses $MWUA$ over its n actions

Player A uses best response, which is to pick σ_B^t that minimses $P_{\sigma_A^t \sigma_B^t} = (\sigma_A^t)^\top r_t$, where r

fill in, if necessary, from online notes

except major corrections in eq above eq. 4 - has x,y wrong way round

this is actually an algorithm for approximating LPs with error poly in $1/\varepsilon$ (though not log thereof)

in proof of MWUA, note that $\boldsymbol{l}_t$ is not a vector of ones, but the loss vector