Learning Half spaces

Definitions

$$\mathcal{C}_{n,3} = \{x \in \{0,1,-1\}^n \mid |\{i \mid x_i \neq 0\}| \leq 3\}$$

$$\mathcal{H}_{n,3} = \{h_{w,b} : C_{n,3} \mapsto \{\pm 1\} \mid h_{w,b}(x) = sign(\langle w, x\rangle + b), w \in R^n, b \in R\}$$

$$Err_D(h) = Pr_{(x,y)\sim D}(h(x) \neq y)$$

$$Err_D(\mathcal{H}) = \min_{h\in\mathcal{H}} Err_D(h)$$

Learning algorithm (for half-spaces):

A learning algoritm L maps samples to hypothesis. In the context of this paper the learning algorithm L, maps training sets/samples as follows:

$$L : (C_{n,3} \times \{\pm 1\})^m \mapsto \mathcal{H}_{n,3}$$

Notice that the output $L(S)$ of the learning algorithm is a hypothesis $\mathcal{H}_{n,3}$, i.e:

$$L(S) \in \mathcal{H}_{n,3}$$

We say L learns $\mathcal{H}_{n,3}$ if for every distribution D on $C_{n,3} \times \{\pm 1\}$ and samples S of more than $m(n,\epsilon)$ i.i.d. examples from from D:

$$Pr_S[Err_D(L(S) > Err_D(\mathcal{H}_{n,3}) + \epsilon] < \frac{1}{10}$$

We say that the learning algorithm is efficient if L returns a hypothesis in $poly(m(n,\epsilon))$ and the hypothesis can be evaluated in polynomial time.

A n-variable 3CNF clause is a boolean formula of the form:

$$C(x) = (-1)^{j_1}x_{i_1} \vee (-1)^{j_2}x_{i_2} \vee (-1)^{j_3}x_{i_3}$$

A 3CNF formula is a boolean formula of the form:

$$\phi(x) = \wedge_{i=1}^{m} C_i(x)$$

To denote this we use $3CNF_{n,m}$ when it has n variables and m clauses.

Let $Val(\phi)$ denote the maximal fraction of clauses that can be simultaneously satisfied.

If $Val(\phi) = 1$ then we say that ϕ is satisfiable.

Boolean formulas can be trivially transformed to formulas with $\{\pm\}$ instead of $\{0,1\}$ and majority operations. First the the majority function defined as follows:

$$\forall(x_1,x_2,x_3) \in \{\pm 1\}^3, MAJ(x_1,x_2,x_3) := sign(x_1+x_2+x_3)$$

An n-variable 3CNF clauses C can be mapped to 3 majority (3MAJ) clauses using the formula:

$$C(x) = MAJ((-1)^{j_1}x_{i_1}, (-1)^{j_2}x_{i_2}, (-1)^{j_3}x_{i_3})$$

An n-variable 3CNF formulas ϕ can be equivalently be expressed using 3MAJ formulas as follow:

$$\phi(x) = \wedge_{i=1}^{m} C_i(x) = \Pi_{i=1}^{m} C_i(x)$$

To denote this we use $3MAJ_{n,m}$ when it has n variables and m clauses.

Conjecture 2.2: (μ-R3SAT hardness assumption) $\forall\epsilon > 0, \forall\Delta > \Delta_o(\epsilon)$, there exists no efficient algorithm that ϵ-refutes random 3CNF with ratio $\Delta \cdot n^{\mu}$.

Theorem 3.1 : Let $0 \leq \mu \leq 0.5$. If the μ-R3SAT hardness assumption (conjecture 2.2) is true, then there exists no efficient learning algorithm that learns the class $\mathcal{H}_{n,3}$ using $O\left(\frac{n^{1+\mu}}{\epsilon^2}\right)$ examples.

To prove theorem 3.1 we will prove a stronger version of it. For that we will need to define:

$$\mathcal{H}_{n,m}^{d} = \{h_{w,0} : C_{n,3} \mapsto \{\pm 1\} \mid h_{w,0}(x) = \langle w, x\rangle, w \in R^n, b = 0\}$$

Notice $\mathcal{H}_{n,m}^{d} \subset \mathcal{H}_{n,m}$, this fact is what makes theorem 3.2 stronger (and hence imply theorem 3.1):

Theorem 3.2 : Under μ-R3SAT hardness assumption, it is impossible to efficiently learn this subclass $\mathcal{H}_{n,m}^{d}$, using only $O\left(\frac{n^{1+\mu}}{\epsilon^2}\right)$.

Proof Sketch:

To show that its impossible to learn $\mathcal{H}_{n,m}^{d}$ (using only $O\left(\frac{n^{1+\mu}}{\epsilon^2}\right)$), we will reduce the problem of ϵ-refuting $3MAJ$ formulas to the problem of learning $\mathcal{H}_{n,m}^{d}$ with $O\left(\frac{n^{1+\mu}}{\epsilon^2}\right)$. With this reduction, we will be able to show that if such a learning algorithm L existed, then we could learn $\mathcal{H}_{n,m}^{d}$ and hence, construct an algorithm that is able to ϵ-refute $3MAJ$ formulas efficiently.

For this reduction to work we will need the following steps:

step1) For this reduction, we will map every 3MAJ clause to two examples in $C_{n,3} \times \{\pm 1\}$. Since 3MAJ are just linear combinations of boolen values, we will just indicate the coefficients in the $x_k \in C_{n,3}$ vector. More precisely, for every clause 3MAJ clause $C(x) = MAJ((-1)^{j_1} x_{i_1}, (-1)^{j_2} x_{i_2}, (-1)^{j_3} x_{i_3})$ one can map it to an example $(x_k, y_k) \in C_{n,3} \times \{\pm 1\}$ by choosing $b \in \{\pm 1\}$ (at random) and letting:

$$(x_k, y_k) = b(\sum_{l=1}^{3} (-1)^{j_l} e_{i_l}, 1) \in (C_{n,3} \times \{\pm 1\})$$

where e_i are the usual standard basis vectors. Conceptually, we are simply using the indices of the boolean vector take part of the current 3MAJ formula to denote the non-zero relevant entries in the vector x_k. The vector y_k is intended to indicate if the current clause is satisfied or not.

step2) Apart from mapping each clause $C(x)$ we will also map every possible $w \in \mathcal{H}^d_{n,m}$ to a possible (boolean) assignment ψ to the 3MAJ formula $\phi(x)$. To do this we will take advantage that $w \in \{\pm 1\}^n$ and that there is a bijection with vectors $w \in \{\pm 1\}^n$ to hyperplanes in $\mathcal{H}^d_{n,m}$.

For this proof to work the following fact is crucial:

step3 Claim: If $\psi \in \{\pm 1\}^n$ and its corresponding hypothesis are $h_{\psi,0}(x) = sign(\langle \psi, x \rangle)$, then $h_{\psi,0}(x_k) = y_k$ if and only if ψ satisfies C_k.

Sketch proof:

This fact is nearly immediate because we constructed x_k to be the "coefficients" of the majority formula but without the actual boolean values x. Therefore one can appreciate that the inner product $\langle \psi, x \rangle$ simply linearly combines the coefficients of 3MAJ (encoded in x_k) and with the satisfying assigning $\psi \in \{\pm 1\}^n$. Since y_k is always flipped depending whether b is 1 or -1, y_k matches $\langle \psi, x \rangle$ if and only if $C(x)$ is satisfied.

step4 Given $\phi \in 3MAJ_{n,\Delta n^{1+\mu}}$ (and for large enough Δ) consisting of 3MAJ clauses $C_1, ..., C_{\Delta n^{1+\mu}}$ we will create a sample set S consisting of $\Delta n^{1+\mu}$ examples (x_k, y_k) for each clause C_k as described in step 2. Now given these samples we will choose a random subset S_1 of size $O(\frac{n^{1+\mu}}{\epsilon})$. Let the empirical distribution induced by choosing $(C_{n,3} \times \{\pm 1\})$ from ϕ be D. Let the learned hypothesis be denoted by $L(S)$.

Now with these ingredients we can use the learned hypothesis $L(S)$ to, to see what fraction of clauses are actually satisfiable or not (i.e. which are random). This is equivalent to constructing an algorithm A that reliable refutes ϕ formulas.

If ϕ is nearly satisfiable, i.e. $Val(\phi) \geq 1-\epsilon$, then, most of the clauses that were mapped to (x_k, y_k) will have a vector x_k that is satisfiable and hence, matches y_k. If this is the case then $L(S)$ will get most of its predictions correct (with high probability, since our learning algorithm is PAC learnable and only learns successfully with high probability). Therefore $Err_D(L(S))$ will be small w.h.p. If this is the case our algorithm will return "exceptional" (and its likely to be satisfiable).

On the other hand, if ϕ is random, then no algorithm can learn ϕ. Why? Well, if ϕ is random then for every (x_k, y_k), y_k will be a Bernoulli r.v. with parameter $\frac{1}{2}$, independent of x_k. Since the instances is basically random and the algorithm was only provided with $O(\frac{n^{1+\mu}}{\epsilon})$, the learning algorithm will not see most of the clauses and therefore, since its seeing something that is random by change, will produce a hypothesis that is independent of its observations. Since these clauses are random, h is likely to make a mistake on about half of the clauses. Therefore, $Err_D(L(S))$ will be close to 1/2 (considered "large", i.e. it makes a large mistake). Therefore, output "typical".

Therefore, in summary, if $Err_D(L(S))$ is large output "typical", otherwise, if the error $Err_D(L(S))$ is small output "exceptional".

Therefore, we have constructed an efficient algorithm with small sample complexity that ϵ-refutes 3MAJ formulas, which should not be possible under conjecture 2.2.