Learning Linear Threshold Function With Noise

ML Learning Theory Final Project

Weiting Tan Jiawei Peng Yi Zhou Yang Xiao
Johns Hopkins University
{wtan12, jpeng30, yzhou188, yxiao40}@jhu.edu

Abstract

In this project, we focus on natural semi-random input models such as Random Classification Noise (RCN) model [1] and its generalization, Massart Noise model [2]. First we follow previous work[3] to prove why halfspaces under RCN model is efficiently PAC learnable. Then, We follow the recent work [2] to discuss how to efficiently PAC learn halfspaces with Massart Noise.

1 Background and Motivation

It's known that in the realizable setting, halfspaces can be efficiently PAC learned with linear programming while it's not efficiently PAC learnable in agnostic setting. Therefore it's worth studying how much we can loose the constraints from realizable setting. Semi-random input models fall into this category where we have an function that randomly corrupts the distribution with probability smaller than 0.5. In this work, we survey related work and give an overview of the approach that can PAC learn the class of halfspaces in the presence of noisy labels. We firstly deal with the Random Classification Noise model, and then work on its generalization, Massart Noise model.

2 Problem Formulation

Halfspaces In this project, we focus on binary classification problem with halfspaces. Halfspaces is defined as the class of Boolean functions: $\mathcal{H} = \{x \mapsto \text{sign}(\langle \vec{w}, x \rangle - \theta)\}$ where $\vec{w} \in \mathbb{R}^d$ is the weight and $\theta \in \mathbb{R}$ is the threshold. It is also known as: Linear Threshold Functions, Perceptrons, Linear Separators, Threshold Gates, Weighted Voting Games, etc.

Random Classification Noise (RCN) Model Following [3], We investigate the setting where distribution is corrupted with a noise example oracle $EX^{RCN}(f, D_x, \eta)$ where D_x is an arbitrary distribution over X and $0 \leq \eta < \frac{1}{2}$. We construct noised distribution D by randomly sampling $x \sim D_x$ then label x as $f(x)$ with probability $1 - \eta$ and label x as $-f(x)$ with probability η, where f(x) is an unknown halfspace.

Massart Noise Model Following [2], Massart Noise model is a generalization of RCN, where we label x as $f(x)$ with probability $1 - \eta(x)$, and label x as $-f(x)$ with probability $\eta(x)$ with $\eta(x) \leq \eta \forall \eta(x)$. Note that $\eta(x)$ is unknown to the learner compared to the fixed η in RCN and is therefore more challenging to devise efficient algorithm.

3 Learning Linear Threshold Functions

We begin our discussion with learning linear threshold functions (halfspaces) without the present of noise. There are different approaches to learn halfspaces in polynomial time and we follow the discussion from [3] and use Perceptron Algorithm as the base algorithm because it is shown to have noise-tolerant property and is easy to be extended and used for learning halfspaces with RCN model. In this section we show what is Perceptron Algorithm and see how to modify it to guarantee poly-time learnability of halfspaces for both clean and noisy inputs. The author uses $\theta = 0$ as the threshold for simplicity and we will follow this construction and only discuss learning homogeneous halfspaces $\mathcal{H} = \{x \mapsto \text{sign}(\langle \vec{w}, x \rangle)\}$

3.1 Perceptron Algorithm

Algorithm 1 Perceptron Algorithm

Denote: $\widehat{x} = \frac{x}{|x|}$
Initiate: w=$\vec{0}$, $S \sim D^m, S_{correct} = \{(x,y)|y(w \cdot x) \geq 0\}, S_{incorrect} = \{(x,y)|y(w \cdot x) < 0\}$
while $S_{incorrect} \neq \emptyset$ **do**
    Randomly Sample $(x,y) \in S_{incorrect}$
    $w \leftarrow w + y(\widehat{x})$
end while

Theorem 3.1. *[4] Assume the set S is realizable (we can find some unit vector w^* that correctly labels all data), the Perceptron Algorithm converges in at most $1/\sigma^2$ iterations where $\sigma = min_{x \in S}|w^* \widehat{x}|$*

Proof. Consider the cosine between current vector w and w^* that we want to find: $\frac{w \cdot w^*}{|w|}$. In each step of the algorithm 1 we have $(w + y(\widehat{x})) \cdot w^* = w \cdot w^* + y\widehat{x} \cdot w^* \geq w \cdot w^* + \sigma$. We also have $|w + y(\widehat{x})|^2 \leq |w|^2 + 1$ since x is classified incorrectly. Therefore after t iteration we have $w \cdot w^* \geq t\sigma, |w|^2 \leq t \implies t\sigma \leq \sqrt{t} \implies t \leq 1/\sigma^2$ □

3.2 Modified Perceptron Algorithm

We have shown Perceptron Algorithm along with theorem that proves its convergence property. However, the "separation parameter" σ from Theorem 3.1 can be exponentially small which results in non-polytime runtime for the Perceptron Algorithm. Thus following [3], we discuss a modified version of Perceptron Algorithm that bound number of iteration in polynomial time.

Algorithm 2 Modified Perceptron Algorithm

Denote: $\widehat{x} = \frac{x}{|x|}$
Initiate: w=$\vec{0}$, $t = 0$
$S \sim D^m, S_{correct} = \{(x,y)|y(w \cdot x) \geq 0\}, S_{incorrect} = \{(x,y)|y(w \cdot x) < 0\}$
while True **do**
    if $\forall x \in S_{incorrect}, |cos(w,x)| \leq \sigma$ **then**
        HALT (ideal w^* found)
    end if
    pick the misclassified $x \in S$ that maximizes $|cos(\omega, x)|$ and update ω using:
    $w \leftarrow w - (w \cdot \widehat{x})\widehat{x}$
    $t = t + 1$
    if $t > \frac{\ln n}{\sigma^2}$ **then**
        $w = \vec{0}$ (Restart the training) $t = 0$
    end if
end while

Theorem 3.2. *If dataset S is realizable, then w.p $\geq 1 - \delta$ the modified perceptron algorithm halts after $O(\frac{\ln(n)\ln(1/\delta)}{\sigma^2})$ iterations and produce a vector $\tilde{w}$ s.t. $\forall x \in S_{incorrect}, |cos(\tilde{w}, x)| \leq \sigma$*

Proof. The proof is similar to proof 3.1 (of standard perceptron algorithm). Let ω^* be a unit vector that correctly classifies all $x \in S$. Suppose it is the case that the initial (random unit) vector ω satisfies $\omega \cdot \omega^* \geq \frac{1}{\sqrt{n}}$. Note that in each update in line (8), $\omega \cdot \omega^*$ does not decrease because: $(w - (w \cdot \widehat{x})\widehat{x}) \cdot \omega^* = \omega \cdot \omega^* - (\omega \cdot \widehat{x})(\omega^* \cdot \widehat{x}) \geq \omega \cdot \omega^*$, where the last inequality holds because ω misclassifies x. On the other hand, we can prove $|\omega|^2$ does decrease significantly using Pythagorean Theorem: $|(w - (w \cdot \widehat{x})\widehat{x})|^2 = |\omega|^2 - 2(\omega \cdot \widehat{x})^2 + (\omega \cdot \widehat{x})^2 \leq |\omega|^2(1 - \sigma^2)$.

Therefore, after t iterations, $|\omega|^2 \leq (1 - \sigma^2)^{\frac{t}{2}}$. Meanwhile, $|\omega| \geq \omega \cdot \omega^*$, which means $t \leq \frac{\ln n}{\sigma^2}$. □

After describing the modified perceptron algorithm, we haven't addressed the issue that σ can be exponentially small. To bound σ, we use the main result from [3]:

Lemma 3.3. ***Outlier Removal Lemma:*** *For any set $S \subseteq I_b^n$ and any $\epsilon > 0$, there exists a subset $S' \subseteq S$ that:*

1. $|S'| \geq (1 - \epsilon - 2^{-nb})|S|$, *and*
2. $\forall w \in R^n, max\_x \in S'(w \cdot x)^2 \leq \beta E_{x \in S'}\left[(w \cdot x)^2\right]$

Where I_b^n limits the input to be b bits of precision: $I_b = \{p/q : |p|, |q| \in \{0, 1, \cdots, 2^b - 1\}, q \neq 0\}$, $I_b^n = (I_b \times I_b \cdots \times I_b)$ and $\beta = O(\frac{n^7 b}{\epsilon})$.

Proof. The proof is sophisticated, here we only sketch the basic idea. For each iteration, we perform linear transformation on input so that, for each unit vector w, $E_{x \in S}\left[(w \cdot x)^2\right] = 1$. Let our input form matrix X, this transformation is done with $A^{-1}X$ where A^2 is symmetric factorization of XX^T (We can define $E = \{x \in R^n : (w^T x)^2 \leq w^T A^2 w\}$ and show it's an ellipsoid). Then we remove all points $x \in S$ that $|x|^2 > \beta/144n$ and check if the remaining points satisfies second criterion of the theorem. If so, we halt, otherwise we repeat the process on new sample set. Note that the tricky part is to prove the remaining $|S'|$ is non-trivial. To do so, we need the following lemma:

Lemma 3.4. *Let μ be a measure on ^{n} which is not concentrated on a subspace of dimension less than n. Then for any $0 < \alpha < 1/3n, \beta = 36n^3/\alpha$, there exists an ellipsoid $S \subseteq^n$ such that*

1. $P(x \notin S) \leq \alpha$
2. *Either*
   (a) *for all $w \in^n, max\{(w^T x)^2 : x \in S\} \leq \beta E[(w^T x)^2 | x \in S]$ or*
   (b) $vol(W(S)) \geq 2vol(W(^n))$

We can show that after applying Lemma 3.4 $k \leq K_0(poly(n, b))$ times, we end up with an intersection of ellipses that satisfies the outlier removal lemma. For the full proof, we refer the readers to check the full proof from [3]. □

With Outlier Removal Lemma, for any sample data S, we can find a non-trivial fraction of S and form a new dataset T such that for every hyperplane defined by vector $w \in R^n$, the maximum distance from a point in T to the plane is at most polynomially larger than the average distance of all points to the plane. Now use Outlier Removal Lemma with modified perceptron algorithm, we can first transform sample dataset S into S' such that for all vector $w, max_{x \in S'}(w \cdot x)^2 \leq \beta E_{S'}\left[(w \cdot x)^2\right]$.Then we linearly transform S' by $A^{-1}X$ shown in the proof and the transformed space satisfies that $E_{S'}\left[(w \cdot x)^2\right] = 1$. Thus for any vector w,

$$\begin{aligned} E_{S'}\left[cos(w, x)^2\right] &= E_{S'}\frac{(w \cdot x)^2}{|x|^2} \\ &\geq \frac{E_{S'}\left[(w \cdot x)^2\right]}{max_{S'}|x|^2} \\ &\geq \frac{1}{\beta n} \end{aligned} \tag{1}$$

Therefore, by this construction, at least $\frac{1}{2\beta n}$ fraction of points in S' satisfies $cos(w,x)^2 \geq \frac{1}{2\beta n}$. Thus we simply set $\sigma = 1/\sqrt{2\beta n}$ for modified perceptron algorithm and after $O(\frac{\ln(n)\ln(1/\delta)}{\sigma^2})$ iterations we have at least $\frac{1}{2\beta n}$fraction of points in S' satisfies $cos(w,x)^2 \geq \frac{1}{2\beta n} = \sigma$. For these points, the learned weight $\tilde{w}$ can correctly classify them with high probability (per Theorem 3.2). Thus we can simply boost this process by running it multiple times, each time at least $\frac{1}{\beta n}$ fraction of input sample data are classfied correctly with high probability. Note that since separation parameter $\sigma = 1/\sqrt{2\beta n}$ and $\beta = O(\frac{n^7 b}{\epsilon})$, $\sigma = poly(n, b, \epsilon)$ which ensures that modified perceptron algorithm will converge in polynomial time.

3.3 Learning with Noisy Input

We have presented an algorithm that PAC learn linear threshold functions in polynomial time with perceptron algorithm. Now we consider the noisy input case where each labeled data is corrupted by some function. Let S^η be the corrupted dataset seen in training and S be the original set of clean samples. For a given vector w, we define

$$S^\eta_{error} = \{x \in S^\eta : cos(w,x) < -\sigma\}$$
$$S^\eta_{correct} = \{x \in S^\eta : cos(w,x) > \sigma\}$$

and claim that $x_{update} = \sum_{x \in S^\eta_{error}} x + \frac{\eta}{1-\eta} \sum_{x \in S^\eta_{correct}} x$ is good enough to perform the update of weight ($w = w - (w \cdot \widehat{x})\widehat{x}$) in modified perceptron training algorithm.

To see why x_{update} is a good vector, we can define

$$S_{error} = \{x \in S : cos(w,x) < -\sigma\}$$
$$S_{correct} = \{x \in S : cos(w,x) > \sigma\}$$

. With respect to the random choice noisy examples, we can have:

$$\mathbf{E}[Sum[S^\eta_{error}]] = (1-\eta)Sum[S_{error}] - \eta Sum[S_{correct}]$$
$$\mathbf{E}[Sum[S^\eta_{correct}]] = (1-\eta)Sum[S_{correct}] - \eta Sum[S_{error}]$$

Then, the expected value of x_{update} is:

$$\begin{aligned}\mathbf{E}[x_{update}] &= \frac{1-\eta-\eta^2}{1-\eta} Sum[S_{error}] \\ &= \frac{1-2\eta}{1-\eta} Sum[S_{error}]\end{aligned}$$

We claim that with high probability, $|x_{update} - E[x_{update}]| \leq \sqrt{m}\log n$ and can show that with such bound, the modified perceptron algorithm's convergence property is still satisfied by considering two cases:

1. x_{update} is small implies that current hypothesis is a good classifier
2. value of x_{update} is sufficiently close to its expectation so that the relaxed conditions are satisfied for modified perceptron algorithm to converge in polynomial time. The relaxed condition are:

$$\begin{aligned} cos(w,x)y &\leq -\sigma/2 \\ cos(w^*,x)y &\geq \frac{-\sigma^2}{16\sqrt{n}\ln n} \end{aligned} \tag{2}$$

Once this is proven, we have shown that modified perceptron algorithm is noise tolerant with respect to a known η (we need η to compute x_{update}).

To prove the convergence property under these two cases, we are going to assume that

$$m = |S_{correct} \cup S_{error}| \geq m_0 = \frac{10^4 n^2 (\ln(n))^4}{\epsilon^2 \delta^6} \frac{(1-\eta)^2}{(1-2\eta)^2}$$

where $\epsilon < \frac{1}{2}$ to derive $|S| \geq 2\beta m_0 n$ according to Lemma 1. Therefore, it's easy to see that $|x_{update} - E[x_{update}]| \leq \sqrt{m}\log n$.

For case 1: if x_{update} is small enough to satisfy $|x_{update}| \leq \frac{1-2\eta}{1-\eta} m\epsilon\delta - \sqrt{m}logn$, then according to the claim we gave, $Sum[S_{error}] \leq m\epsilon\delta$. Further, by definition, since each point in S satisfies that $|x| \geq cos(x, \omega) \geq \delta$, we can get $|Sum[S_{error}]| > \delta|S_{error}|$. Therefore, $|S_{error}| \leq m\epsilon$ and we have achieved the goal of having a good classifier.

For case 2: if x_{update} is large and satisfies $|x_{update}| > \frac{1-2\eta}{1-\eta} m\epsilon\delta - \sqrt{m}logn$. Then with our claim, we know that with high probability,

$$\begin{aligned}\frac{\omega^* \cdot x_{update}}{|x_{update}|} &\geq \frac{\omega^* \cdot \mathbf{E}[x_{update}]}{|x_{update}|} - \frac{\sqrt{m}logn}{|x_{update}|} \\ &\geq \frac{-\sqrt{m}logn}{|x_{update}|}\end{aligned}$$

and

$$\begin{aligned}\mathbf{E}(\omega \cdot x_{update}) &= (\frac{1-2\eta}{1-\eta})\omega \cdot Sum[S_{error}] \\ &\leq -(\frac{1-2\eta}{1-\eta})\delta|S_{error}||\omega| \\ &\leq -\delta \cdot \frac{\omega}{2}\end{aligned}$$

which implies (2).

For the general case where η is unknown, we can perform the algorithm with respect to guessed η in $\{0, 1/|S|, 2/|S|, \cdots, 1/2\}$ and evaluate these models performance. Note that this method does not work when η is not a fixed value (namely in the Massart Noise model case). Thus, in the following sections, we follow the recent advance on Massart Noise [2] and present a polynomial time algorithm to learn halfspaces with Massart Noise Model.

4 Learning Linear Threshold Functions with Massart Noise Model

Next, we will explore the problem of PAC learning independent of the distribution of halfspaces with Massart noise by interpreting [2]. The author starts with large margin case which is easy for the RCN model since the problem will be essentially convex. That is, there is a convex surrogate that allows us to formulate the problem as a convex program. We can use SGD to find a near-optimal solution to this convex program, which automatically gives a strong proper learner. Unfortunately, it doesn't work is for Massart noise and the author shows in Theorem 3.1 [2] that no convex surrogate can lead to a weak learner, even under a margin assumption.

However, the author proves Lemma 2.5 [2] which establishes that *even though minimizing a convex proxy does not lead to small misclassification error over the entire space, there exists a region with non-trivial probability mass where it does.* This lemma is the main novelty and the key contribution of this paper. Motivated by this, the author comes up with an algorithm that adaptively applies a sequence of convex optimization problems to obtain an accurate solution in disjoint subsets of the space.

4.1 Some Necessary Notations

- $\mathcal{D}_{\mathbf{x}}$: the marginal of $\mathcal{D}$ on $\mathbf{x}$, $\mathcal{D}_y(\mathbf{x})$: the distribution of y conditional on $\mathbf{x}$.
- Convex proxy for the misclassification error: $L(\mathbf{w}) = \mathbf{E}_{(\mathbf{x},y)\sim\mathcal{D}}[\text{LeakyRelu}_\lambda(-y\langle\mathbf{w}, \mathbf{x}\rangle)]$, under the constraint $\|\mathbf{w}\|_2 \leq 1$, where $\text{LeakyRelu}_\lambda(z) = \begin{cases}(1-\lambda)z & \text{if } z \geq 0 \\ \lambda z & \text{if } z < 0\end{cases}$ and λ is the leakage parameter, which we will set to be $\lambda \approx \eta$.
- Per-point misclassification error: $err(\mathbf{w}, \mathbf{x}) = \mathbf{Pr}_{y\sim\mathcal{D}_y(\mathbf{x})}[\mathbf{w}(\mathbf{x}) \neq y]$
- Error of the proxy function: $\ell(\mathbf{w}, \mathbf{x}) = \mathbf{E}_{y\sim\mathcal{D}_y(\mathbf{x})}[\text{LeakyRelu}_\lambda(-y\langle\mathbf{w}, \mathbf{x}\rangle)]$.
- OPT $= \min_{h\in\mathcal{H}_d} err_{0-1}^{\mathcal{D}}(h)$ denotes the optimal misclassification error of any halfspace, and $\mathbf{w}^*$ is the normal vector to the optimal halfspace.

- Note that $\mathrm{err}_{0-1}^{\mathcal{D}}(h_{\mathbf{w}}) = \mathbf{E}_{x\sim\mathcal{D}_x}[\mathrm{err}(\mathbf{w},\mathbf{x})]$ and $L(\mathbf{w}) = \mathbf{E}_{x\sim\mathcal{D}_x}[\ell(\mathbf{w},\mathbf{x})]$. Moreover, OPT$=$ $\mathbf{E}_{x\sim\mathcal{D}_x}[\mathrm{err}(\mathbf{w}^*,\mathbf{x})] = \mathbf{E}_{x\sim\mathcal{D}_x}[\eta(x)]$

4.2 Proof of Lemma 2.5

4.2.1 Prerequisite

Claim 2.1 [2]: *For any* $\mathbf{w},\mathbf{x}$*, we have that* $\ell(\mathbf{w},\mathbf{x}) = (err(\mathbf{w},\mathbf{x}-\lambda)|\langle\mathbf{w},\mathbf{x}\rangle|$.

Proof. We consider two cases:
(1) **Case** $\mathrm{sign}(\langle\mathbf{w},\mathbf{x}\rangle) = \mathrm{sign}(\langle\mathbf{w}^*,\mathbf{x}\rangle)$: In this case, we have that $err(\mathbf{w},\mathbf{x}) = \eta(\mathbf{x})$, while $\ell(\mathbf{w},\mathbf{x}) = \eta(\mathbf{x})(1-\lambda)|\langle\mathbf{w},\mathbf{x}\rangle| - (1-\eta(\mathbf{x}))\lambda|\langle\mathbf{w},\mathbf{x}\rangle| = (\eta(\mathbf{x})-\lambda)|\langle\mathbf{w},\mathbf{x}\rangle|$.
(2) **Case** $\mathrm{sign}(\langle\mathbf{w},\mathbf{x}\rangle) \neq \mathrm{sign}(\langle\mathbf{w}^*,\mathbf{x}\rangle)$: In this case, we have that $err(\mathbf{w},\mathbf{x}) = 1-\eta(\mathbf{x})$, while $\ell(\mathbf{w},\mathbf{x}) = (1-\eta(\mathbf{x}))(1-\lambda)|\langle\mathbf{w},\mathbf{x}\rangle| - \eta(\mathbf{x})\lambda|\langle\mathbf{w},\mathbf{x}\rangle| = (1-\eta(\mathbf{x})-\lambda)|\langle\mathbf{w},\mathbf{x}\rangle|$. □

Claim 2.1 finds the relationship between LeakyReLU Loss and Margin, and it is also used for the proof of the following lemmas.

Lemma 2.3[2] :*If* $\lambda \geq \eta$*, then* $L(\mathbf{w}^*) \geq -\gamma(\lambda - \mathrm{OPT})$

Proof. For any fixed x, using Claim 2.1, we have that

$$\ell(\mathbf{w}^*,\mathbf{x}) = (err(\mathbf{w}^*,\mathbf{x})-\lambda)|\langle\mathbf{w}^*,\mathbf{x}\rangle| = (\eta(x)-\lambda)|\langle\mathbf{w}^*,\mathbf{x}\rangle| \leq -\gamma(\lambda-\eta(x)), \tag{3}$$

since $|\langle\mathbf{w}^*,\mathbf{x}\rangle| \geq \gamma$ and $\eta(x)-\lambda \leq 0$. Taking expectation over $x\sim\mathcal{D}_x$, the statement follows. □

Lemma 2.3 is used to ensure that the assumption of Lemma 2.5 (negative optimal loss) is established, as long as λ is appropriately selected.

Lemma 2.4[2] : *Let* L *be any convex function. Consider the (projected) SGD iteration htat is initialized at* $\mathbf{w}^{(0)} = \mathbf{0}$ *and for every step computes*

$$\mathbf{w}^{(t+\frac{1}{2})} = \mathbf{w}^{(t)} - \rho\mathbf{v}^{(t)} \quad \textit{and} \quad \mathbf{w}^{(t+1)} = arg\min_{\mathbf{w}:\|\mathbf{w}\|\leq 1} \|\mathbf{w}-\mathbf{w}^{(t+\frac{1}{2})}\|_2$$

, where $\mathbf{v}^{(t)}$ *is a stochastic gradient such that for all steps* $\mathbf{E}[\mathbf{v}^{(i)}|\mathbf{w}^{(i)}] \in \partial L(\mathbf{w}^{(t)})$ *and* $\|\mathbf{v}^{(t)}\|_2 \leq 1$*. Assume that SGD is run for* T *iterations with step size* $\rho = \frac{1}{\sqrt{T}}$ *and let* $\bar{\mathbf{w}} = \frac{1}{T}\sum_{t=1}^{T}\mathbf{w}^{(t)}$*. Then, for any* $\epsilon,\delta > 0$*, after* $T = \Omega(\log(1/\delta)/\epsilon^2)$ *iterations with probability at least* $1-\delta$*, we have that* $L(\bar{\mathbf{w}}) \leq \min_{\mathbf{w}:\|\mathbf{w}\|\leq 1} L(\mathbf{w}) + \epsilon$.

Lemma 2.4 is mainly used to ensure that an approximate optimal solution can be found in Lemma 2.3. To be more specific, we know that $\min_{\mathbf{w}:\|\mathbf{w}\|\leq 1} L(\mathbf{w}) \leq -\gamma(\lambda - \mathrm{OPT})$ according to Lemma 2.3. By Lemma 2.4, it follows that by running SGD on $L(\mathbf{w})$ with projection to the unit ℓ_2-ball for $O(\log(1/\delta)/(\gamma^2(\lambda-\mathrm{OPT})^2))$ steps, we find a $\mathbf{w}$ such that $\min_{\mathbf{w}:\|\mathbf{w}\|\leq 1} L(\mathbf{w}) \leq -\gamma(\lambda-\mathrm{OPT})/2$ with probability at least $1-\delta$.

4.2.2 Lemma 2.5

Lemma 2.5 is the core lemma of this article, two conditions guarantee that we can divide the probability space, and use the idea of "Divide and Conquer" to reduce the error.

Lemma 2.5[2] :*Consider a vector w with* $L(\mathbf{w}) < 0$*. There exists a threshold* $T \geq 0$ *such that* (1)$\mathbf{Pr}_{(x,y)\sim\mathcal{D}}[|\langle\mathbf{w},\mathbf{x}\rangle| \geq T] \geq \frac{|L(\mathbf{w})|}{2\lambda}$*, and (2)*$\mathbf{Pr}_{(x,y)\sim\mathcal{D}}[h_w(x) \neq y|\langle\mathbf{w},\mathbf{x}\rangle \geq T] \leq \lambda - \frac{L(\mathbf{w})}{2}$.

Proof. Show there is a $T \geq 0$ such that $\mathbf{Pr}_{(x,y)\sim\mathcal{D}}[h_w(x) \neq y|\langle\mathbf{w},\mathbf{x}\rangle \geq T] \leq \lambda - \zeta$, where $\zeta \stackrel{\mathrm{def}}{=} \frac{L(\mathbf{w})}{2}$, or equivalently, $\mathbf{E}_{x\sim\mathcal{D}_x}[(err(\mathbf{w},\mathbf{x})-\lambda+\zeta)\mathbb{1}_{|\langle\mathbf{w},\mathbf{x}\rangle|\geq T}] \leq 0$.

For a T drawn uniformly at random in [0,1], we have that:

$$\int_0^1 \mathop{\mathbf{E}}_{x\sim\mathcal{D}_x}[(err(\mathbf{w},\mathbf{x})-\lambda+\zeta)\mathbb{1}_{|\langle\mathbf{w},\mathbf{x}\rangle|\geq T}]dT = \mathbf{E}_{x\sim\mathcal{D}_x}[(err(\mathbf{w},\mathbf{x})-\lambda)|\langle\mathbf{w},\mathbf{x}\rangle|] + \zeta\mathbf{E}_{x\sim\mathcal{D}_x}[|\langle\mathbf{w},\mathbf{x}\rangle|]$$
$$\leq \mathbf{E}_{x\sim\mathcal{D}_x}[\ell(\mathbf{w},\mathbf{x})] + \zeta = L(\mathbf{w}) + \zeta = L(\mathbf{w})/2 < 0, \tag{4}$$

where the first inequality uses Claim 2.1. Thus, there exists a $\bar{T}$ such that

$$\mathbf{E}_{x\sim\mathcal{D}_x}[(err(\mathbf{w},\mathbf{x})-\lambda+\zeta)\mathbb{1}_{|\langle\mathbf{w},\mathbf{x}\rangle|\geq\bar{T}}] \leq 0. \tag{5}$$

Consider the minimum such $\bar{T}$. Then we have

$$\int_{\bar{T}}^1 \mathbf{E}_{x\sim\mathcal{D}_x}[(err(\mathbf{w},\mathbf{x})-\lambda+\zeta)\mathbb{1}_{|\langle\mathbf{w},\mathbf{x}\rangle|\geq T}]dT \geq \lambda\cdot\mathbf{Pr}_{(x,y)\sim\mathcal{D}}[|\langle\mathbf{w},\mathbf{x}\rangle|\geq\bar{T}]. \tag{6}$$

By definition of $\bar{T}$, it must be the case that

$$\int_0^{\bar{T}} \mathbf{E}_{x\sim\mathcal{D}_x}[(err(\mathbf{w},\mathbf{x})-\lambda+\zeta)\mathbb{1}_{|\langle\mathbf{w},\mathbf{x}\rangle|\geq T}]dT \geq 0. \tag{7}$$

Therefore,

$$\frac{L(\mathbf{w})}{2} \geq \int_{\bar{T}}^1 \mathbf{E}_{x\sim\mathcal{D}_x}[(err(\mathbf{w},\mathbf{x})-\lambda+\zeta)\mathbb{1}_{|\langle\mathbf{w},\mathbf{x}\rangle|\geq T}]dT \geq \lambda\cdot\mathbf{Pr}_{(x,y)\sim\mathcal{D}}[|\langle\mathbf{w},\mathbf{x}\rangle|\geq\bar{T}]. \tag{8}$$

which implies that $\mathbf{Pr}_{(x,y)\sim\mathcal{D}}[|\langle\mathbf{w},\mathbf{x}\rangle|\geq\bar{T}] \geq \frac{|L(\mathbf{w})|}{2\lambda}$. This completes the proof of Lemma 2.5. □

From Lemma 2.5, we can see that:

- As long as we reduce the loss function to a sufficiently large negative value (by Lemma 2.3), T is guaranteed to be found, where a sample space with larger measure and a margin greater than T can be divided.
- The expected loss is smaller in this space. (less than $\lambda = \eta + \epsilon$)

4.3 Large Margin Case

We consider the case that there is no probability mass within distance γ from the separating hyperplane $\langle\mathbf{w}^*,\mathbf{x}\rangle = 0$, $\|\mathbf{w}^*\|_2 = 1$.Then assume that for every $\mathbf{x}\sim\mathcal{D}_\mathbf{x}$, $\|\mathbf{x}\|_2 \leq 1$ and that $|\langle\mathbf{w}^*,\mathbf{x}\rangle| \geq \gamma$.

Algorithm 3 Algorithm with Margin

Set: $S^{(1)} = \mathbb{R}^d, \lambda = \eta+\epsilon, m = \tilde{O}(\frac{1}{\gamma^2\epsilon^4})$
Set: $i \leftarrow 1$
Draw $O((\frac{1}{\epsilon^2})\log(1/(\epsilon\gamma))$ samples from $\mathcal{D}_\mathbf{x}$ to form an empirical distribution $\tilde{\mathcal{D}}_\mathbf{x}$.
while $\mathbf{Pr}_{\mathbf{x}\sim\tilde{\mathcal{D}}_\mathbf{x}}[\mathbf{x}\in S^{(i)}] \geq \epsilon$ **do**
  Set $\mathcal{D}^{(i)} = \mathcal{D}|_{S^{(i)}}$, the distribution conditional on the unclassified points.
  Let $L^{(i)}(\mathbf{w}) = \mathbf{E}_{(x,y)\sim\mathcal{D}^{(i)}}[\text{LeakyRelu}_\lambda(-y\langle\mathbf{w},\mathbf{x}\rangle)]$
  Run SGD on $L^{(i)}(\mathbf{w})$ for $\tilde{O}(\frac{1}{\gamma^2\epsilon^2})$ iterations to get $\mathbf{w}^{(i)}$ with $\|\mathbf{w}^{(i)}\|_2 = 1$ such that $L^{(i)}(\mathbf{w}^{(i)}) \leq \min_{\mathbf{w}:\|\mathbf{w}\|_2\leq 1} L^{(i)}(\mathbf{w}) + \gamma\epsilon/2$
  Draw m samples from $\mathcal{D}^{(i)}$ to form an empirical distribution $\mathcal{D}_m^{(i)}$
  Find a threshold $T^{(i)}$ such that $\mathbf{Pr}_{(\mathbf{x},y)\sim\mathcal{D}_m^{(i)}}[|\langle\mathbf{w}^{(i)},\mathbf{x}\rangle| \geq T^{(i)}] \geq \gamma\epsilon$ and the empirical misclassification error, $\mathbf{Pr}_{(\mathbf{x},y)\sim\mathcal{D}_m^{(i)}}[h_{\mathbf{w}^{(i)}}(\mathbf{x}) \neq y || \langle\mathbf{w}^{(i)},\mathbf{x}\rangle| \geq T^{(i)}]$, is minimized.
  Update the unclassified region $S^{(i+1)} \leftarrow S^{(i)} \setminus \{\mathbf{x} : |\langle\mathbf{w}^{(i)},\mathbf{x}\rangle| \geq T^{(i)}\}$ and set $i \leftarrow i+1$
end while
Return the classifier $[(\mathbf{w}^{(1)},T^{(1)}),(\mathbf{w}^{(2)},T^{(2)}),...]$

Figure 1 shows the relationship between the theorems and algorithms design. Note the **DKW Inequality[5]** is

$$Pr(\sup_{x\in\mathbb{R}}|F_n(x) - F(x)| > \epsilon) \leq Ce^{-2m\epsilon^2} \quad \text{for every } \epsilon > 0$$

According to it and the empirical distribution, removal of the measure of $\gamma\epsilon$ on sample $m = \tilde{O}(\frac{1}{\gamma^2\epsilon^4})$ can ensure that the measure of $\gamma\epsilon$ is removed from the true distribution.

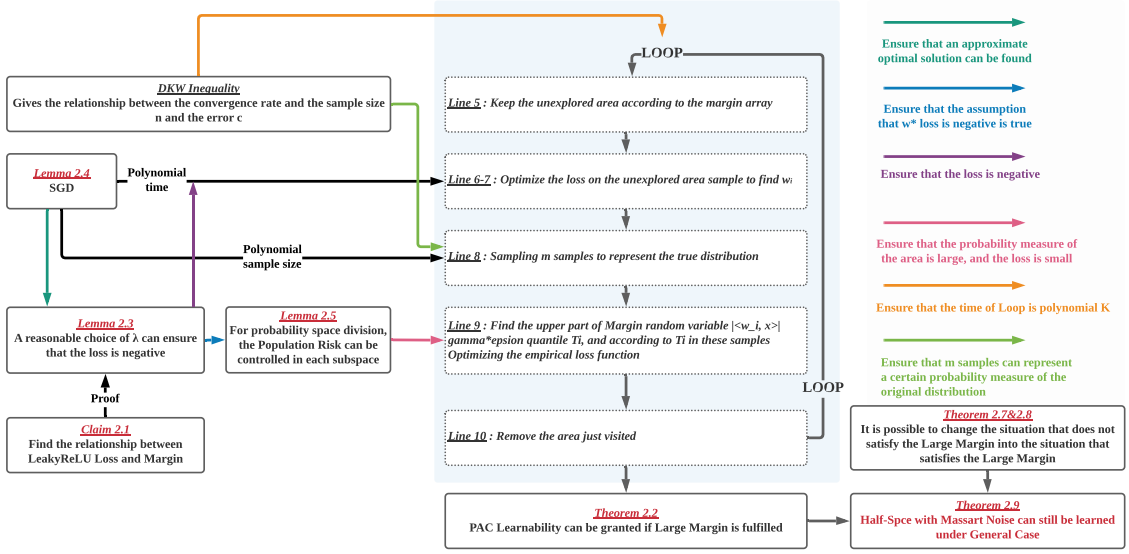

Figure 1: Relation Between the Theorems (as described in Algorithm 3).

4.4 Extended to General Case

In the general case, we assume that $\mathcal{D}_{\mathbf{x}}$ is an arbitrary distribution supported on b-bit integers. While such a distribution might have exponentially small margin in the dimension d (or even 0), we will preprocess the distribution to ensure a margin condition by removing outliers.

Definition 2.7[2, 6] : We call a point x in the support of a distribution$\mathcal{D}_x$ a β-outlier, if there exists a vector $w \in \mathbb{R}^d$ such that $\langle w, x\rangle^2 \geq \beta \mathbf{E}_{x\sim\mathcal{D}_X}[\langle w, x\rangle^2]$.

Lemma 2.8 [2, 6]: *Using $m = \tilde{O}(d^2b)$ samples from $\mathcal{D}_{\mathbf{x}}$, one can identify with high probability an ellipsoid E such that $\mathbf{Pr}_{\mathbf{x}\sim\mathcal{D}_{\mathbf{x}}}[\mathbf{x} \in E] \geq \frac{1}{2}$ and $\mathcal{D}_{\mathbf{x}|E}$ has no $\Gamma^{-1} = \tilde{O}(db)$-outliers.*

Lemma 2.8 shows that any distribution supported on b-bit integers can be efficiently preprocessed using samples so that no large outliers exist. Hence, we can adapt Algorithm 3 for the large margin case to work in general.

Algorithm 4 Algorithm for General Case

Set: $S^{(1)} = \mathbb{R}^d, \lambda = \eta + \epsilon, \Gamma^{-1} = \tilde{O}(db), m = \tilde{O}(\frac{1}{\Gamma^2\epsilon^4})$
Set: $i \leftarrow 1$
Draw $O((\frac{1}{\epsilon^2})\log(1/(\epsilon\Gamma))$ samples from $\mathcal{D}_{\mathbf{x}}$ to form an empirical distribution $\tilde{\mathcal{D}}_{\mathbf{x}}$.
while $\mathbf{Pr}_{\mathbf{x}\sim\tilde{\mathcal{D}}_{\mathbf{x}}}[\mathbf{x} \in S^{(i)}] \geq \epsilon$ **do**
    Run the algorithm of Lemma 2.8 to remove Γ^{-1}-outliers from the distribution $\mathcal{D}_{S^{(i)}}$ by filtering points outside the ellipsoid $E^{(i)}$
    Let $\Sigma^{(i)} = \mathbf{E}_{(\mathbf{x},y)\sim\mathcal{D}^{(i)}|_{S^{(i)}}}[\mathbf{x}\mathbf{x}^T]$ and set $\mathcal{D}^{(i)} = \Gamma\Sigma^{(i)-1/2} \cdot \mathcal{D}|_{S^{(i)}\cap E^{(i)}}$ be the distribution $\mathcal{D}|_{S^{(i)}\cap E^{(i)}}$ brought in isotropic position and rescaled by Γ so that all vectors have ℓ_2-norm at most 1.
    Let $L^{(i)}(\mathbf{w}) = \mathbf{E}_{(\mathbf{x},y)\sim\mathcal{D}^{(i)}}[\text{LeakyRelu}_\lambda(-y\langle\mathbf{w},\mathbf{x}\rangle)]$
    Run SGD on $L^{(i)}(\mathbf{w})$ for $\tilde{O}(\frac{1}{\Gamma^2\epsilon^2})$ iterations, to get $\mathbf{w}^{(i)}$ with $\|\mathbf{w}^{(i)}\|_2 = 1$ such that $L^{(i)}(\mathbf{w}^{(i)}) \leq \min_{\mathbf{w}:\|\mathbf{w}\|_2\leq 1} L^{(i)}(\mathbf{w}) + \Gamma\epsilon/2$
    Draw m samples from $\mathcal{D}^{(i)}$ to form an empirical distribution $\mathcal{D}_m^{(i)}$
    Find a threshold $T^{(i)}$ such that $\mathbf{Pr}_{(\mathbf{x},y)\sim\mathcal{D}_m^{(i)}}[|\langle\mathbf{w}^{(i)},\mathbf{x}\rangle| \geq T^{(i)}] \geq \Gamma\epsilon$ and the empirical misclassification error, $\mathbf{Pr}_{(\mathbf{x},y)\sim\mathcal{D}_m^{(i)}}[h_{\mathbf{w}}(\mathbf{x}) \neq y||\langle\mathbf{w}^{(i)},\mathbf{x}\rangle| \geq T^{(i)}]$, is minimized.
    Revert the linear transformation by setting $\mathbf{w}^{(i)} \leftarrow \Gamma\Sigma^{(i)-1/2}\mathbf{w}^{(i)}$.
    Update the unclassified region $S^{(i+1)} \leftarrow S^{(i)} \setminus \{\mathbf{x} : \mathbf{x} \in E^{(i)} \wedge |\langle\mathbf{w}^{(i)},\mathbf{x}\rangle| \geq T^{(i)}\}$ and set $i \leftarrow i + 1$
end while
Return the classifier $[(\mathbf{w}^{(1)}, T^{(1)}, E^{(1)}), (\mathbf{w}^{(2)}, T^{(2)}, E^{(2)}), ...]$

4.5 Main Result

Theorem 2.2[2] :(Large margin)*Let $\mathcal{D}$ be a distribution on $\mathbb{B}_d \times \{\pm 1\}$ such that $\mathcal{D}_x$ satisfies that γ-margin property with respect to w^* and y is generated by sign $(\langle\mathbf{w}^*,\mathbf{x}\rangle)$ corrupted with Massart noice at rate $\eta < \frac{1}{2}$. Algorithm 3 uses $\tilde{O}(\frac{1}{\gamma^3\epsilon^5})$samples from $\mathcal{D}$, runs in poly$(d, \frac{1}{\epsilon}, \frac{1}{\gamma})$ time, and returns, with probability $\frac{2}{3}$, a classifier h with misclassification error* $\text{err}_{0-1}^{\mathcal{D}}(h) \leq \eta + \epsilon$

According to the definition 2.7 and Lemma 2.8, this method can convert the problem that does not satisfy γ-Margin into the problem that satisfies γ-Margin, thus solving the PAC-learning proof of General Case.

Theorem 2.9[2] :(General)*Let $\mathcal{D}$ be a distribution over (d+1)-dimensional labeled examples with bit-complexity b, generated by an unknown halfspace corrupted by Massart noise at rate $\eta < \frac{1}{2}$. Algorithm 4 uses $\tilde{O}(\frac{d^3b^3}{\epsilon^5})$ samples, runs in poly $(d, \frac{1}{\epsilon}, b)$ time, and returns, with probability $\frac{2}{3}$, a classifier h with misclassification error*$\text{err}_{0-1}^{\mathcal{D}}(h) \leq \eta + \epsilon$.

4.6 Conclusion

In conclusion, as mentioned by the author[2],the key idea of this paper is to use the Large Margin hypothesis to get the Lemma 2.5. Also, it is possible to deduce a suitable loss function to make Lemma 2.5 hold. According to Lemma 2.5, the idea of "Divide and Conquer" can be used to divide the entire probability space, and ensure that the Population Risk of each subspace is small, and it keeps looping.

References

[1] Dana Angluin and Philip D. Laird. Learning from noisy examples. *Machine Learning*, 2:343–370, 1988.

[2] Ilias Diakonikolas, Themis Gouleakis, and Christos Tzamos. Distribution-independent pac learning of halfspaces with massart noise, 2019.

[3] A. Blum, A. Frieze, R. Kannan, and S. Vempala. A polynomial-time algorithm for learning noisy linear threshold functions. In *Proceedings of 37th Conference on Foundations of Computer Science*, pages 330–338, 1996.

[4] Marvin Minsky and Seymour Papert. Cambridge, MA, USA.

[5] A. Dvoretzky, J. Kiefer, and J. Wolfowitz. Asymptotic Minimax Character of the Sample Distribution Function and of the Classical Multinomial Estimator. *The Annals of Mathematical Statistics*, 27(3):642 – 669, 1956.

[6] John Dunagan and Santosh Vempala. Optimal outlier removal in high-dimensional spaces. *Journal of Computer and System Sciences*, 68(2):335–373, 2004.