Assignment 2
Advanced Machine Learning

Armand Nicolicioiu (AI - 407)

September 2021

Exercise 1 Let $\mathcal{X}$ be an instance space. The learning algorithm A is better than the learning algorithm B with respect to some probability distribution, D, if we have:

$$L_D(A(S)) \leq L_D(B(S))$$

for all samples $S \in (\mathcal{X} \times \{0,1\})^m$. Prove that for every distribution $\mathcal{D}$ over $\mathcal{X} \times \{0,1\}$ there exists a learning algorithm A_D that is better than any other algorithm with respect to $\mathcal{D}$.

Solution:
Let A_i and S_j be all the possible learning algorithms and all the possible training sets, respectively.
Out of all the algorithms, $\exists A_{i^*}$ and $\exists S_{j^*}$ such that the hypothesis h_* resulted when training A_{i^*} on the sample set S_{j^*} obtains the lowest generalization error possible (out of all the algorithms A_i applied on a training set S_j).

Let B be an algorithm such that, for a given training set, S outputs:

$$B = \begin{cases} A_{i^*}(S_{j^*}), \text{if } S \neq S_{j^*} \text{ (ignores its input } S) \\ A_{i^*}(S), \text{if } S = S_{j^*} \end{cases}$$

In other words, $B(S) = A_{i^*}(S_{j^*})$. Thus, we have a learning algorithm B that is better than any other algorithm with respect to $\mathcal{D}$.

Consider the Bayes predictor:

$$f_{\mathcal{D}} = \begin{cases} 1, \text{if } \mathbb{P}[y=1|x] \geq 1/2 \\ 0, \text{otherwise} \end{cases}$$

We know (proved in the seminar) that for any probability distribution $\mathcal{D}$, the Bayes predictor $f_{\mathcal{D}}$ is optimal, in the sens that no other classifier $g : \mathcal{X} \to \{0,1\}$ has a lower error, $L_{\mathcal{D}}(f_{\mathcal{D}}) \leq L_{\mathcal{D}}(g)$.
If it is possible to output the Bayes optimal predictor for our distribution $\mathcal{D}$ with a learning algorithm, then this algorithm is better than any other algorithm with respect to $\mathcal{D}$.

Exercise 2 Show that the class $\mathcal{H}$ of concentric circles centered in origin in the 2D plane can be (ϵ, δ) - PAC learned by giving an algorithm A and determining an upper bound on the sample complexity $m_{\mathcal{H}}(\epsilon, \delta)$ such that the definition of PAC-learnability is satisfied. Compute VCdim($\mathcal{H}$).

Solution:
The hypothesis $\mathcal{H}$ is (ϵ, δ) - PAC learnable if for every $\epsilon, \delta > 0$, for every labeling function $f \in \mathcal{H}^2_{rec}$ (realizability case) and for every distribution $\mathcal{D}$ on $\mathbb{R}^2$ we can find a function $m_{\mathcal{H}} \colon (0,1)^2 \to \mathbb{N}$ and a learning algorithm A such that, when ran on a training set S consisting of $m \geq m_{\mathcal{H}}(\epsilon, \delta)$ examples sampled i.i.d. from $\mathcal{D}$ and labeled by f, the algorithm A returns a hypothesis $h_S \in \mathcal{H}$ that has a real risk lower than ϵ with probability at least $1 - \delta$.

$$\begin{aligned} &\underset{S \sim \mathcal{D}^m}{P}(L_{f,\mathcal{D}}(h_S) \leq \epsilon) \geq 1 - \delta \\ \implies &\underset{S \sim \mathcal{D}^m}{P}(L_{f,\mathcal{D}}(h_S) > \epsilon) < \delta \end{aligned}$$

We will first define the algorithm A.
We consider the training set $S = \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$.
We find the point x_S that is furthest away from the origin, from the set S_+ of points with label 1.

$$r_S = \max_{x_i \in S_+} ||x_i||_2 \tag{1}$$

We define the hyphotheis $h_S = C(r_S) = C_S$ the circle of radius equal to the distance from the origin to the point r_S. By construction, A is an ERM algorithm with $L_{h^*,\mathcal{D}}(h_S) = 0$ because h_S doesn't mislable any example in the training set.

From the realizability assumption, there exists a hypothesis $h^* = C(r^*) = C^*$ such that all the points inside the circle $C(r^*)$ have $1(+)$ as the correct label, and all the points outside it have $0(-)$ as the correct label.

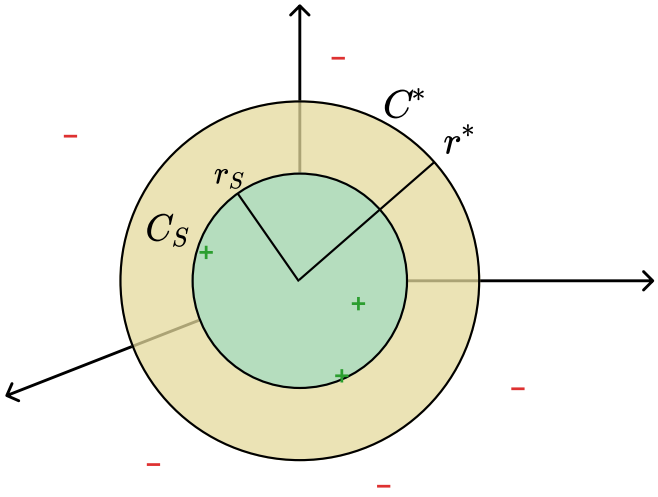

Figure 1

Now we want to find the sample complexity $m_{\mathcal{H}}(\epsilon, \delta)$ such that

$$\underset{S\sim\mathcal{D}^m}{P}(L_{h^*,\mathcal{D}}(h_S) \leq \epsilon) \geq 1 - \delta \text{ where } S \text{ contains } m \geq m_{\mathcal{H}}(\epsilon, \delta) \text{ examples.}$$

Because of the realizability assumption and our of choice of A, it is assured that h_S will label correctly any point inside C_S. Also, we make the observation that $C_S \subset C^*$, meaning that h_S also labels correctly all the examples outside C^* because they are outside C_S too. Thus, the only region of space where h_S can make mistakes is $C^* \setminus C_S$.

Case 1)

$$\begin{aligned} &\mathcal{D}(C^*) = \underset{x\sim\mathcal{D}}{P}(x \in C^*) \leq \epsilon \\ \implies &L_{h^*,\mathcal{D}}(h_S) = \underset{x\sim\mathcal{D}}{P}(h_S(x) \neq h^*(x)) = \underset{x\sim\mathcal{D}}{P}(x \in C^* \setminus C_S) \leq \underset{x\sim\mathcal{D}}{P}(x \in C^*) \leq \epsilon \\ \implies &\underset{S\sim\mathcal{D}^m}{P}(L_{h^*,\mathcal{D}}(h_S) \leq \epsilon) = 1 \text{ (this happens all the time)} \end{aligned}$$

Case 2)

$$\mathcal{D}(C^*) = \underset{x\sim\mathcal{D}}{P}(x \in C^*) > \epsilon \tag{2}$$

We define $T_\epsilon = C^* \setminus C_\epsilon = C(r^*) \setminus C(r_\epsilon)$ is a region of space close to the margins of the C^* such that $\underset{x\sim\mathcal{D}}{P}(x \in T_\epsilon) = \epsilon$ (see Figure 2)

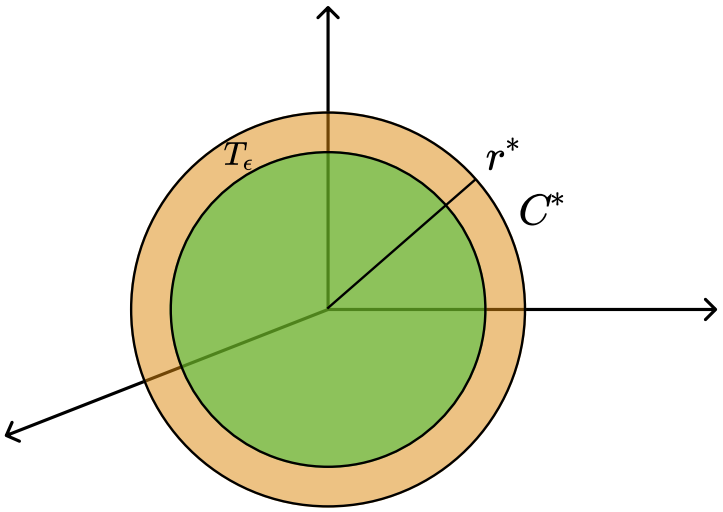

Figure 2: T_ϵ (orange color) represents the region at the margin of the C^* ball.

Case 2.1) C_S intersects T_ϵ, results:

$$L_{h^*,\mathcal{D}}(h_S) = \underset{x\sim\mathcal{D}}{P}(h^*(x) \neq h_S(x)) = \underset{x\sim\mathcal{D}}{P}(x \in C^* \setminus C_S) \leq \underset{x\sim\mathcal{D}}{P}(x \in T_\epsilon) = \epsilon \quad \text{(with probability 1)}$$

Case 2.2) $C_S \cap T_\epsilon = \emptyset$

$$\underset{S\sim\mathcal{D}^m}{P}(L_{h^*,\mathcal{D}}(h_S) > \epsilon) = \underset{S\sim\mathcal{D}^m}{P}(C_S \cap T_\epsilon = \emptyset)$$

$\underset{S\sim\mathcal{D}^m}{P}(C_S \cap T_\epsilon = \emptyset)$ is the probability that no point from T_ϵ is sampled in the training set S. This probability is $(1-\epsilon)^m$. We obtain:

$$\underset{S\sim\mathcal{D}^m}{P}(L_{h^*,\mathcal{D}}(h_S) > \epsilon) = \underset{S\sim\mathcal{D}^m}{P}(C_S \cap T_\epsilon = \emptyset) = (1-\epsilon)^m \leq e^{-\epsilon m}$$

$$\begin{aligned} & e^{-\epsilon m} < \delta \\ \implies & -\epsilon m < \log\delta \\ \implies & m > -\frac{1}{\epsilon}\log\delta \\ \implies & m > \frac{1}{\epsilon}\log\frac{1}{\delta} \end{aligned}$$

Therefore, for a training set S of size $m \geq m_{\mathcal{H}}(\epsilon,\delta) = \frac{1}{\epsilon}\log\frac{1}{\delta}$ i.i.d. samples from $\mathcal{D}$, our learning algorithm A obtains a hypothesis h_S with $\underset{S\sim\mathcal{D}^m}{P}(L_{h^*,\mathcal{D}}(h_S) \leq \epsilon) \geq 1-\delta$, showing that $\mathcal{H}$ is (ϵ,δ) - PAC learnable. □

b. VCdim($\mathcal{H}$)

It is easy to see that any set of one point $X = \{x\}$ is shattered by $\mathcal{H}$. We denote $d = ||x||_2$ the distance from the point x to the origin. The two possible labelings can be realised by selecting a circle with radius $r < d$ for label $0(-)$ or selecting a circle with radius $r > d$ for label $1(+)$.

So:

$$\text{VCdim}(gH) \geq 1 \tag{3}$$

Let $X_2 = \{x_1, x_2\}$ a set of two points. We will show that $\mathcal{H}$ cannot shatter X_2.

Without loosing generality, we can sort any two points such that the first one is closer to the origin than the second point (or at equal distance) $||x_1||_2 \leq ||x_2||_2$.

Let's assume that the labeling (0,1) is realised by a hyphothesis $h = C(r)$. The second point x_2 has label 1(+), so:

$$x_2 \in C(r) \implies ||x_2|| < r$$
$$||x_1||_2 \leq ||x_2||_2 < r \implies x_1 \in C(r)$$

Thus, the first point x_1 also has label 1(+), contradicting the assumption. Thus, the labeling (0,1) cannot be realised for any set of two points, which means that any such set X_2 cannot be shattered.

This results in

$$\text{VCdim}(gH) < 2$$
$$Eq.3 \implies \text{VCdim}(gH) = 1$$

Exercise 3 Consider the concept class C formed by closed intervals $[a, b]$ with $a, b \in \mathbb{R}$:

$$C = \{h_{a,b} : \mathbb{R} \to \{0, 1\}, a \leq b, h_{a,b}(x) = \mathbf{1}_{[a,b]}(x)\}$$

Compute the shattering coefficient $\tau_H(m)$ of the growth functior for $m \geq 0$.

Solution:

From the definition, for a hypothesis class $\mathcal{H}$, $\tau_H(m)$ is the maximum number of different functions from a set C of size m to $\{0, 1\}$ that can be obtained by restricting $\mathcal{H}$ to C.

$$\tau_H(m) = \max_{C \subseteq X : |C| = m} |\mathcal{H}_C|$$

Let $X = \{x_1, x_2, ..., x_m\}$ be a set of m points. Without losing generality, consider $x_i < x_j$.

The concept class C assigns label 1 to a single continuous region (i.e a subarray) of our sorted array of points.

For the base case, we have no positive label:

$$(0, 0, 0, \ldots, 0)$$

Now, we count the number of labelings with i positive points, for $i = \overline{1, m}$.

$$i = 1 \implies \{(1, 0, 0, \ldots, 0), (0, 1, 0, \ldots, 0), \ldots, (0, 0, 0, \ldots, 1)\} \ m \text{ labelings.}$$
$$i = 2 \implies \{(1, 1, 0, \ldots, 0), (0, 1, 1, 0, \ldots, 0), \ldots, (0, \ldots, 0, 1, 1)\} \ m - 1 \text{ labelings.}$$
$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$
$$i = m - 1 \implies \{(1, 1, 1, \ldots, 1, 0), \ldots, (0, 1, 1, \ldots, 1, 1)\} \ 2 \text{ labelings.}$$
$$i = m \implies \{(1, 1, 1, \ldots, 1)\} \ 1 \text{ labeling.}$$

In total, we have 1*(the base case with no positive labels)* $+ m + (m - 1) + \cdots + 1$ labelings. Thus,

$$\tau_H(m) = 1 + \frac{m \cdot (m + 1)}{2}$$

Exercise 4 Consider a concept class C_2 formed by union of two closed intervals, that is $[a, b] \cup [c, d]$, with $a, b, c, d \in \mathbb{R}$ (with $a \leq b \leq c \leq d$). Give an efficient ERM algorithm for learning the concept class C_2 and compute its complexity for each of the following cases:

a. realizable case.

b. agnostic case.

Solution:

a. In the realizable case, there exists a function $h_{a^*,b^*,c^*,d^*}(x) = (\mathbf{1}_{[a^*,b^*]} \cup \mathbf{1}_{[c^*,d^*]})(x)$ that labels the training points.

$$S = \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\} \qquad y_i = h_{a^*,b^*,c^*,d^*}(x_i)$$

First, we sort all the points in S in ascentind order according to the $x's$, obtaining:

$$S = \{(x_{\sigma(1)}, y_{\sigma(1)}), (x_{\sigma(2)}, y_{\sigma(2)})(x_{\sigma(m)}, y_{\sigma(m)})\} \text{ with } x_{\sigma(1)} \leq x_{\sigma(2)} \leq \cdots \leq x_{\sigma(m)}.$$

Consider the following ERM algorithm for determining a, b, c, d after sorting the training set S.

1. If there are no positive labels return: $h_{a,b,c,d}$, where $a = x_{\sigma(1)} - 1, b = x_{\sigma(1)} - 1, c = x_{\sigma(m)} + 1, d = x_{\sigma(m)} + 1$.
2. If there are only positive examples return: $h_{a,b,c,d}$ where $a = x_{\sigma(1)}, b = c = d = x_{\sigma(m)}$.
3. if $\exists(\mathrm{x}_i, y_i) \in$S such that $\mathrm{y}_i == 1$ then:
   a = b = $\min_{i=\overline{1,m},\ y_i=+1} x_i$.
   c = d = $\max_{i=\overline{1,m},\ y_i=+1} x_i$.
4. We found the final values for at least a and d, now let's adjust the values for b and c.
   for $i = \overline{2, m}$
   if $(y_{\sigma(i)} == -1$ and $y_{\sigma(i-1)} == +1)$
   then $b = x_{\sigma(i-1)}$
   if $(y_{\sigma(i)} == +1$ and $y_{\sigma(i-1)} == -1)$
   then $c = x_{\sigma(i)}$
   return: $h_{a,b,c,d}$

Let's compute the complexity of this algorithm:

Sorting: $\mathcal{O}(m \cdot \log m)$.
Determining there are no positive labels: $\mathcal{O}(m)$.
Determining there are only positive labels: $\mathcal{O}(m)$.
Final iteration to adjust b and c: $\mathcal{O}(m)$.

Total complexity: $\mathcal{O}(m \cdot \log m)$.

b. In the agnostic case, we can have different labels for the same point, so we are dealing with a distribution $\mathcal{D}$ over $\mathcal{X} \times \{0, 1\}$.

Similarly to the realizable case, we start by sorting the trainins set S according to the $x's$, obtaining:

$$S = \{(x_{\sigma(1)}, y_{\sigma(1)}), (x_{\sigma(2)}, y_{\sigma(2)})(x_{\sigma(m)}, y_{\sigma(m)})\} \text{ with } x_{\sigma(1)} \leq x_{\sigma(2)} \leq \cdots \leq x_{\sigma(m)}.$$

Consider the set Z containing the values of x' with no repetition:

$$Z = \{z_1, z_2, \ldots, z_n\}$$
$$z_1 = x_{\sigma(1)} < z_2 < \cdots < z_n = x_{\sigma(m)} \quad n \leq m$$

1. If all $y_i = 0$, in order to return an ERM algorithm we can pick two intervals outside the training set S, for example $a_S = b_S = c_S = d_S = x_{\sigma(1)} - 1$.

2. Consider all possible two intervals reunions $Z_{i,j,k,l} = [z_i, z_j] \cup [z_k, z_l], i = \overline{1,n}, j = \overline{i,n}, k = \overline{j,n}, l = \overline{k,n}$.

For the ERM algorithm, we have to determine the solution $Z^* = Z_{i^*,j^*,k^*,l^*}$ with the smallest empirical risk. We compute this as:

$$\text{Loss}(Z_{i,j,k,l}) = \frac{\#\text{ negative points inside } Z_{i,j,k,l} + \#\text{ positive points outside } Z_{i,j,k,l}}{m}$$

To make the implementation more efficient, we pre-compute the total number of positive (pos_prefix_i) and negative points (neg_prefix_i) with value $x \leq x_{\sigma(i)}$ using a dynamic programming approach of prefix-sums. Because we can have multiple points with the same value x, we need the auxiliary pos_i and neg_i, the number of points with positive, respectively negative labels and value $x = x_{\sigma(i)}$. Considering the base case $pos\_prefix_0 = neg\_prefix_0 = 0$, we have the recurrence, for $i = \overline{1,n}$:

$$pos\_prefix_i = pos\_prefix_{i-1} + pos_i$$
$$neg\_prefix_i = neg\_prefix_{i-1} + neg_i$$

Now, we fix the limits i, j, k, l and find the ones that minimize $\text{Loss}(Z_{i,j,k,l})$. An efficient ERM algorithm for this is:

1. Sort S and obtain $x_{\sigma(1)} \leq x_{\sigma(2)} \leq \cdots \leq x_{\sigma(m)}$. Build set Z containing value x without repetition: $Z = \{z_1, z_2, \ldots, z_n\},\ z_1 = x_{\sigma(1)} < z_2 < \cdots < z_n = x_{\sigma(m)}$

2. Check if all $y_i\ i = \overline{1,m}$ have value 0. If so, return h_{a_S,b_S,c_S,d_S}, where $a_S = b_S = c_S = d_S = z_1 - 1$.

3. for $j = \overline{1,n}$

   compute values $pos_j =$ # points $x_i = z_j$ with label $y_i = 1$
   $neg_j =$ # points $x_i = z_j$ with label $y_i = 0$

4. for $i = \overline{1,n}$

   $pos\_prefix_i = pos\_prefix_{i-1} + pos_i$
   $neg\_prefix_i = neg\_prefix_{i-1} + neg_i$

5. min_error $= \frac{m}{m} = 1$, $i^* = []$, $j^* = []$, $k^* = []$, $l^* = []$.

   for $i = \overline{1,n}$
   for $j = \overline{i,n}$
   for $k = \overline{j,n}$
   for $l = \overline{k,n}$

$$\text{Loss}(Z_{i,j,k,l}) = \frac{(neg\_prefix_j - neg\_prefix_{i-1}) + (neg\_prefix_l - neg\_prefix_{k-1})}{m} + \frac{pos\_prefix_n - (pos\_prefix_j - pos\_prefix_{i-1}) - (pos\_prefix_l - pos\_prefix_{k-1})}{m}$$

   if $\text{Loss}(Z_{i,j,k,l}) <$ min_error
   min_error $= \text{Loss}(Z_{i,j,k,l})$
   $i^* = i$
   $j^* = j$
   $k^* = k$
   $l^* = l$

6. return h_{a_S,b_S,c_S,d_S}, where $a_S = z_{i^*}, b_S = z_{j^*}, c_S = z_{k^*}, d_S = z_{l^*}$.

Let's compute the complexity of this algorithm:

1. Sorting: $\mathcal{O}(m \cdot \log m)$.
2. Linear check: $\mathcal{O}(m)$.
3. Auxiliary counts pre-compute: $\mathcal{O}(m)$.
4. Prefix-sums DP pre-compute: $\mathcal{O}(m)$.
5. Finding best i, j, k, l combination: $\mathcal{O}(m^4)$ (constant time for Loss using pre-computed prefix-sums).

Total complexity: $\mathcal{O}(m^4)$.

Exercise 5 Consider H^d_{2DNF} the class of 2-term disjunctive normal form formulae consisting of hypothesis of the form $h : \{0,1\}^d \to \{0,1\}$,

$$h(x) = A_1(x) \vee A_2(x)$$

where $A_i(x)$ is a Boolean conjunction of literals (in H^d_{conj}).

It is known that the class H^d_{2DNF} is not efficient properly learnable but can be learned improperly considering the class H^d_{2CNF}. Give a γ-weak-learner algorithm for learning the class H^d_{2DNF} which is not a stronger PAC algorithm for H^d_{2DNF} (like the one considering H^d_{2CNF}). Prove that this algorithm is a γ-weak-learner algorithm for H^d_{2DNF}.

Solution:

From the definition, a learning algorithm A is a γ-weak-learner for a class $\mathcal{H}$ if there exists a function $m_{\mathcal{H}} : (0,1) \to \mathbb{N}$ such that: for every $\delta > 0$, for every labeling function $f \in \mathcal{H}, f : \mathcal{X} \to \{-1, +1\}$, for every distribution $\mathcal{D}$ over $\mathcal{X}$, when we run the learning algorithm A on a training set, consisting of $m > m_{\mathcal{H}}(\delta)$ examples sampled i.i.d from $\mathcal{D}$ and labeled by f, the algorithm A return a hypothesis h (h might not be from $\mathcal{H}$ - improper learning) such that, with probability at least $1-\delta$ (over the choice of examples), $L_{\mathcal{D},f} \leq 1/2-\gamma$.

Using the distribution rule, we can transform the 2-term DNF formula to a 2-CNF formula:

$$A_1 \vee A_2 = \bigwedge_{u \in A_1, v \in A_2} (u \vee v) = \bigwedge_{u \in A_1, v \in A_2} y_{u,v}$$

With this, we obtain a conjunction of $(2n)^2$ variables, each of them being a disjunction of 2 literals from the original problem. This conjunction can be efficiently PAC learned.

To make the problem even simpler, we want a learning algorithm A_{weak} that "drops" one variable from the conjunction (say, without losing generality, $y_{u,v}$) and learns to predict the conjunction of the remaining y type variables. We will show that this is a γ-weak-learner for a class H^d_{2DNF}.

By removing the variable $y_{u,v} = (u \vee v)$ from the conjunction, it is equivalent with saying that we assume its value is $+1$. The chance to be mistaken (both u and v are -1) is $1/4$ ($1/2 \times 1/2$, both literals having 50% chance to be positive.)

Now, we return the answer of the conjunction problem for the remaining of the $(2n)^2 - 1$ variables, having introduced an additional $1/4$ chance of error to the accuracy of this predictor.

We know that C_n, the concept class of conjunctions of at most n Boolean literals is PAC learnable with sample complexity $m_{\mathcal{H}}(\epsilon, \delta) = \left\lceil \frac{1}{\epsilon}(nlog(3) - log(\delta)) \right\rceil$.

This results in: $L_{\mathcal{D}}(A_{weak}(S)) \leq 1/4$(introduced by the variable elimination) $+ \epsilon$.

We want $L_{\mathcal{D}}(A_{weak}(S)) < 1/2 - \gamma$.

Take ϵ such that $1/4 + \epsilon < 1/2 \implies \epsilon < 1/4$. Then, the ERM algorithm for solving the resulted conjunction of Booleans (with one "dropped" variable) is a γ-weak-learner for a class H^d_{2DNF}, where $\gamma = \epsilon$.

Going back to the definition, we proved that for every $\delta > 0$, for every labeling function $f \in \mathcal{H}, f : \mathcal{X} \to \{-1, +1\}$, for every distribution $\mathcal{D}$ over $\mathcal{X}$, we have the function $m_{\mathcal{H}}(\gamma, \delta) = \left\lceil \frac{1}{\gamma}((2n)^2 log(3) - log(\delta)) \right\rceil$ such that when we run the algorithm A_{weak} on $m > m_{\mathcal{H}}(\delta)$ examples sampled i.i.d from $\mathcal{D}$ and labeled by f, the algorithm A_{weak} return a hypothesis h (h might not be from $\mathcal{H}$ - improper learning) such that, with probability at least $1 - \delta$ (over the choice of examples), $L_{\mathcal{D},f} \leq 1/2 - \gamma$. (for γ values $< 1/4$). Thus, our A_{weak} algorithm is a γ-weak-learner algorithm for learning the class H^d_{2DNF}.