Assignment 2 - Solutions

Badea Adrian Catalin, 407

June 20, 2022

1. **(1.5 points)** Consider $\mathcal{H}$ the class of 3-piece classifiers (signed intervals):

$$\mathcal{H} = \{h_{a,b,s} : \mathbb{R} \to \{-1, 1\} \mid a \leq b, s \in \{-1, 1\}\},\ where\ h_{a,b,s}(x) = \begin{cases} s, & x \in [a, b] \\ -s, & x \notin [a, b] \end{cases}$$

a. Compute the shattering coefficient $\tau_H(m)$ of the growth function for $m \geq 0$ for hypothesis class $\mathcal{H}$. **(1 point)**

b. Compare your result with the general upper bound for the growth functions and show that $\tau_H(m)$ obtained at previous point a is not equal with the upper bound. **(0.25 points)**

c. Does there exist a hypothesis class $\mathcal{H}$ for which is equal to the general upper bound (over or another domain $\mathcal{X}$)? If your answer is yes please provide an example, if your answer is no please provide a justification. **(0.25 points)**

Solution

a. First, we have to know e $VCdim(\mathcal{H})$. In seminar 3's solutions, it has been demonstrated that $VCdim(\mathcal{H}) = 3$ in the following manner:

Considering $C = \{x_1, x_2, x_3\}$ a set of 3 distinct points with $x_1 < x_2 < x_3$, it has been shown that $\mathcal{H}$ shatters C, by outputting any set of labels, so $VCdim(\mathcal{H}) \geq 3$. (*)

Also, $\mathcal{H}$ doesn't shatter the following labels $(-1, 1, -1, 1)$, for any C with $|C| = 4$. So $VCdim(\mathcal{H}) < 4$ (**)

From (*) and (**), $VCdim(\mathcal{H}) = 3$

In general, we notice that the hypothesis class can output for a set $C = \{x_1, x_2, x_3..x_n\}$ with $x_1 < x_2 < ..x_n$ and $|C| = m$, only the following type of labels:

- Case 1. $-1, -1, ... - 1, 1, 1..1, -1, ... - 1$ - a sequence of -1s (possibly of null length), followed by a sequence of 1s, and then finishing with a sequence of -1s again, this can be formally translated to $(-1)^p, 1^k, (-1)^{(m-p-k)}$, $p >= 0, k >= 0, p + k <= m$
- Case 2. $1, 1, ..1, -1, -1, .. - 1, 1, ..1, 1$, a sequence of 1s (possibly of null length), followed by a sequence of -1s, finishing with another sequence of 1s, this can be formally translated to $(1)^p, (-1)^k, 1^{(m-p-k)}$, $p >= 0, k >= 0, p + k <= m$

We will count how many possibilities we have for each case. First, we should consider trivial cases for both options, where $k = 0$ or $k = m$.

- In Case 1, if $k = 0$, we have only one possibility : (-1, -1, -1, ... -1). If $k = m$, we have the other trivial case : (1, 1, .. 1)
- In Case 2, if $k = 0$, we have only one possibility : (1, 1, 1, ... 1). If $k = m$, we have the other trivial case : (-1, -1, .. -1)

For this case, we have 2 labelings (i). In these, the value labeled is unique.

Now, we can consider only the solutions for when $0 < k < m$.

If $p = m$, then k would be 0 ($p + k <= m$) , has been discussed previously.

If $p = 0$, then Case 1 becomes $(1, 1, ..1, -1, -1... - 1)$. Case 2 becomes $(-1, -1, ... - 1, 1, 1, ..1)$. Here, we have $m - 1$ labelings (k takes values from 1 to n - 1) for the first case and $m - 1$ labelings for the second case. In total, $2 * m - 2$ labelings for which the value changes only once. (ii)

And now we consider the cases for which $0 < p < m$ and $0 < k < m$. We should note here that if $p + k = m$, then $m - p - k = 0$ and these labelings have already been counted at the last case.

- For the first case, we will discuss all $0 < k < m$, as $k = 0$ or $k = m$ have been already taken into consideration.
  - If $k = 1$, p can take values from 1 to $m - 2$.
  - If $k = 2$, p can take values from 1 to $m - 3$... ...
  - If $k = m - 2$, $p = 1$.
  - If $k = m - 1$, p can take no values

  In total, we have $(m - 2) + (m - 3) + .. + 1 = (m - 2) * (m - 1)/2$
- For the second case, we reach the same number of sequences.

For this case $(m - 2) * (m - 1)$ (iii).

From (i), (ii), and (iii), we get a total sum of $(m-2)*(m-1)+2*m-2+2 = m^2-3*m+2+2*m = m^2 - m + 2$ sequences.

So, the shattering coefficient $\tau_H(m)$ of the growth function for $m \geq 0$ for hypothesis class $\mathcal{H}$ is bounded by $m * (m - 1) + 2$.

b. From Sauer's Lemma, for a hypothesis class, $\mathcal{H}$, with c, and for all m, we have $\tau_H(m) <= \sum_{i=0}^{d} C_m^i$.

We will note the general upper bound as $U = \sum_{i=0}^{d} C_m^i$, thus $\tau_H(m) <= U$.

In our case, when d = 3, the general upper bound is $U = C_m^0 + C_m^1 + C_m^2 + C_m^3 = 1 + m + (m - 1) * m/2 + (m - 2) * (m - 1) * m/6$.

So, $U = 1 + m + (m^2 - m)/2 + (m^2 - 3 * m + 2) * m/6$

$U = 1 + m + m^2/2 - m/2 + m^3/6 - 3 * m^2/6 + 2 * m/6$

$U = m^3/6 + m^2/2 - m^2/2 + m + -m/2 + m/3$

$U = m^3/6 + m - m/2 + m/3$

$U = m^3/6 + 5 * m/6 + 1$

The general upper bound for the growth fuction has degree 3, but our the shattering coefficient $\tau_H(m)$ has degree 2.
We can easily check some cases :

- If $m = 1$, the general upper bound is $1/6 + 5/6 + 1 = 2$
- If $m = 2$, the general upper bound is $8/6 + 10/6 + 1 = 4$
- If $m = 3$, the general upper bound is $27/6 + 15/6 + 1 = 8$
- If $m = 4$, the general upper bound is $64/6 + 20/6 + 1 = 15$. The actual value for the shattering coefficient function here is 14.

These three values for m (1, 2, 3) will be roots for the difference function between the shattering coefficient function and the upper bound.

Let's compare the two functions : $\tau_H(m) \leq U$

$m^2 - m + 2 \leq m^3/6 + 5 * m/6 + 1$

$m^2 - m + 1 \leq m^3/6 + 5 * m/6$

$6 * m^2 - 6 * m + 6 \leq m^3 + 5 * m$

$0 \leq m^3 - 6 * m^2 + 11 * m - 6$

We will define the following function $f: \mathbf{N} \longrightarrow \mathbf{R}$ with $f(m) = m^3 - 6 * m^2 + 11 * m - 6$.

As we know from above, we found the solutions $m = 1, m = 2, m = 3$
So, $f(m) = (m-1) * (m-2) * (m-3)$. We know that for $(m = 1, m = 2, m = 3)$, $f(m) = 0$ and $\tau_H(m) = U$
Also, $f'(m) = 3 * m^2 - 12 * m + 11$, which has solutions for:

$m = 2 - \frac{1}{\sqrt{(3)}}$ and

$m = 2 + \frac{1}{\sqrt{(3)}}$
So, for $m \geq 3$, $f'(m) > 0$, $f(m)$ is increasing and will never be 0 again, as $m = 3$ is biggest root.

In conclusion, we showed that the result from **a.** is different than the upper bound.

c. Yes, there exists a hypothesis class $\mathcal{H}$ for which the general upper bound is equal to the shattering coefficient function.
I will use the hypothesis class $\mathcal{H}_{thresholds}$, where $h_a: \mathbf{R} \longrightarrow \{0, 1\}$ from Lecture 5.

We will compute the shattering coefficient $\tau_H(m)$ for $\mathcal{H}_{thresholds}$.

We notice that for this hypothesis class, the outputs for our set $\mathcal{C}$ that can be obtained are of the following pattern: $(1, 1, 1, ...1, 0, 0, ..0)$. A sequence of 1s followed by a sequence of 0s.

So, our label sets have the form $1^p 0^{m-p}$. The number of 1s (p) can take values from 0 to m - because the number of 0s $(m - p)$, is not less than 0.

The shattering coefficient function $\tau_H(m) = m + 1$

From Lecture 6, we know that $VCdim(\mathcal{H}) = 1$. So, from Sauer's, we get that $\tau_H(m) = m + 1 \leq C_m^0 + C_m^1$. The general upper bound is $m + 1$, which is equal to the shattering coefficient function $\tau_H(m)$.

2. **(1.5 points)** Consider the concept class C_2 formed by the union of two closed intervals $[a, b] \cup [c, d]$, where $a, b, c, d \in \mathbb{R}, a \leq b \leq c \leq d$. 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. **(1 point)**

b. agnostic case. **(0.5 point)**

Solution

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

Let $\{x_1, x_2, ..x_m\}$ be the set of the training points. For each x_i, $y_i = h_{a*,b*,c*,d*}(x)$.

Then, $S = \{(x_1, y_i), (x_2, y_2, ..(x_m, y_m)\}$
We can sort the pairs x_i, y_i in ascending order, depending on x_i. The pairs will be sorted by the permutation σ.
S becomes $\{(x_{\sigma(1)}, y_{\sigma(1)}), (x_{\sigma(2)}, y_{\sigma(2)}, ..(x_{\sigma(m)}, y_{\sigma(m)})\}$, with $x_{\sigma(1)} \leq x_{\sigma(2)} \leq .. \leq x_{\sigma(m)}$.

We can consider the next ERM algorithm for determining a,b,c,d after sorting the training set and continuing with these steps.

0. We can calculate the number of positive examples in O(n).

1.a. If there are only positive examples, we can choose $a = b = c = x_{\sigma(1)}$, and $d = x_{\sigma(m)}$. The output of the function will always be 1, as $\forall x_i \in [c, d]$.

1.b. If there are no positive examples, we can choose $a = b = c = d = x_{\sigma(1)} - 1$.

2. If there are both positive and negative examples, we continue with this step. We can calculate a function z_i with dynamic programming, as being the number of consecutive 1s finishing at i.

We can start with $z_0 = 0$ and then $\forall i = \overline{1, m}$:
if $(y_{\sigma(i)} == 1)$, then $z_i = 1 + z_{i-1}$, else $z_i = 0$.

3. With z_i calculated for $\forall i$ from 1 to m, we can choose the two maxima from this sequence, let them be z_k and z_p and then we can choose for a, b, c, d:

$a = x_{\sigma(k-z_k+1)}$, $b = x_{\sigma(k)}$

$[a, b]$ will include $x_{\sigma(k-z_k+1)}, x_{\sigma(k-z_k+2)}, ..x_{\sigma(k)}$, and we know their y's are 1.
And similarly we will take $c = x_{\sigma(p)}$, $d = x_{\sigma(p-z_p+1)}$

Then, we return $h_{a,b,c,d}$

Let's compute the complexity of the algorithm:
Sorting : $\mathcal{O}(m * logm)$
Calculating the number of positives/negatives : $\mathcal{O}(m)$
In case all are positives/negatives $\mathcal{O}(1)$
Else, Calculating function z : $\mathcal{O}(m)$ and calculating a, b, c, d: $\mathcal{O}(1)$

Total complexity : $\mathcal{O}(m * logm)$.

b. In the agnostic case, it might be the case that there is no labeling function, but instead a distribution.

Similarly to the realizable case, we have $S = \{(x_1, y_i), (x_2, y_2, ..(x_m, y_m)\}$, and we can start by sorting the points.

So, S becomes $\{(x_{\sigma(1)}, y_{\sigma(1)}), (x_{\sigma(2)}, y_{\sigma(2)}, ..(x_{\sigma(m)}, y_{\sigma(m)})\}$, with $x_{\sigma(1)} \leq x_{\sigma(2)} \leq .. \leq x_{\sigma(m)}$.

Consider the set $\mathcal{Z} = \{z_1, z_2, ..z_n\}$ containing the values of x_i without repetitions .

$x_{\sigma(1)} = z1 =< z_2 < z_3 < .. < z_n = x_{\sigma(n)}$, $n \leq m$, n is the number of distinct values of x_i

As we are in the agnostic case, we can have $x_{\sigma(i)} = x_{\sigma(i+1)}$ but $y_{\sigma(i)} \neq y_{\sigma(i+1)}$

i) If all $y_i = 0$, we can pick two intervals outside the training set with $a, b, c, d = z_1 - 1$
ii) If all $y_i = 1$, we can pick contain the whole set in the intervals, with $a = z_1$ and $b, c, d = z_n$.
iii) We can consider all the possible intervals $W_{i,j,k,l} = [z_i, z_j] \bigcup [z_k, z_l]$, $i, j, k, l = \overline{1, n}$

For the ERM algorithm, we have to determine the solution $W^*_{i,j,k,l}$ with the smallest empirical risk. We will compute the loss.

We will note :

PI_W = number of positive points inside $W_{i,j,k,l}$
PO_W = number of positive points outside $W_{i,j,k,l}$

NI_W = number of negative points inside $W_{i,j,k,l}$
NO_W = number negative points outside $W_{i,j,k,l}$

Then, $Loss(W^*) = \frac{PO_W + NI_W}{m}$

Then, we compute pos_i, for $i = \overline{1,n}$, the number of positives with $x = z_i$ and neg_i, for $i = \overline{1,n}$, the number of negatives with $x = z_i$.

In order to make the algorithm more efficient, we can use dynamic programming to pre-compute even the number of positives and negatives with $x \leq z_i$, for $i = \overline{1,n}$.

So, we can compute pos_pre_i and neg_pre_i with the following rules :

$pos\_pre_0 = 0$
$pos\_pre_i = pos\_pre_{i-1} + pos_i$

$neg\_pre_0 = 0$
$neg\_pre_i = neg\_pre_{i-1} + neg_i$

And we know $a \leq b \leq c \leq d$, and we can write the variables defined above for .

$PI_W = pos\_pre_j - pos\_pre_{i-1} + pos\_pre_l - pos\_pre_{k-1}$
$PO_W = pos\_pre_n - PI_W$

$NI_W = neg\_pre_j - neg\_pre_{i-1} + neg\_pre_l - neg\_pre_{k-1}$
$NO_W = neg\_pre_n - NI_W$

Now, we can the limits i, j, k, l and find the best solution that minimizes $Loss(W^*)$. The efficient ERM algorithm is :

0. Sort S and obtain $\{(x_{\sigma(1)}, y_{\sigma(1)}), (x_{\sigma(2)}, y_{\sigma(2)}, ..(x_{\sigma(m)}, y_{\sigma(m)})\}$, with $x_{\sigma(1)} \leq x_{\sigma(2)} \leq .. \leq x_{\sigma(m)}$. Create set $\mathcal{Z}$.

1. For every $i = \overline{1,n}$, calculate pos_i, neg_i.

pos_i = number of points with x value equal to z_i and $y_i = 1$.
neg_i = number of points with x value equal to z_i and $y_i = 0$.

2. With the recurrences discussed above, we can calculate pos_pre_i and neg_pre_i.

3. If $pos\_pre_n = 0$ or $neg\_pre_n = 0$, returns the solutions as discussed above.

4. Start with $min_{error} = 1$ and $i_{sol} = 0, j_{sol} = 0, k_{sol} = 0, l_{sol} = 0$.

For $i = \overline{1,n}$

For $j = \overline{i,n}$

For $k = \overline{j,n}$

For $l = \overline{k,n}$

$Loss(W_{i,j,k,l}) = \frac{PO_W + NI_W}{m} = \frac{pos\_pre_n - (pos\_pre_j - pos\_pre_{i-1} + pos\_pre_l - pos\_pre_{k-1})}{m} + \frac{neg\_pre_j - neg\_pre_{i-1} + neg\_pre_l - neg\_pre_{k-1}}{m}$

If $Loss(W_{i,j,k,l}) < min_{error}$:

$min_{error} = Loss(W_{i,j,k,l})$

$i_{sol} = i, j_{sol} = j, k_{sol} = k, l_{sol} = l$

5. Return $h_{a,b,c,d}$, where $a = z_{i_sol}$, $b = z_{j_sol}$, $c = z_{k_sol}$, $d = z_{l_sol}$.

Complexity of this algorithm:

Step 0 : Sort the set S and obtain Z. $\mathcal{O}(m * logm)$.
Step 1 : Calculate pos_i, neg_i. $\mathcal{O}(m)$.
Step 2 : Calculate pos_pre_i and neg_pre_i. $\mathcal{O}(m)$.
Step 3 : Return solutions in case of unique labels. $\mathcal{O}(1)$.
Step 4 : Finding best i, j, k, l. $\mathcal{O}(m^4)$.
Step 5 : $\mathcal{O}(1)$.
Total complexity : $\mathcal{O}(m^4)$.

3. **(1.5 points)** Consider a modified version of the AdaBoost algorithm that runs for exactly three rounds as follows:

- the first two rounds run exactly as in AdaBoost (at round 1 we obtain distribution $\mathbf{D}^{(1)}$, weak classifier h_1 with error ϵ_1; at round 2 we obtain distribution $\mathbf{D}^{(2)}$, weak classifier h_2 with error ϵ_2).
- in the third round we compute for each $i = 1, 2, \ldots, m$:

$$\mathbf{D}^{(3)}(i) = \begin{cases} \frac{D^{(1)}(i)}{Z}, & if\ h_1(x_i) \neq h_2(x_i) \\ 0, & otherwise \end{cases}$$

  where Z is a normalization factor such that $\mathbf{D}^{(3)}$ is a probability distribution.
- obtain weak classifier h_3 with error ϵ_3.
- output the final classifier $h_{final}(x) = sign(h_1(x) + h_2(x) + h_3(x))$.

Assume that at each round $t = 1, 2, 3$ the weak learner returns a weak classifier h_t for which the error ϵ_t satisfies $\epsilon_t \leq \frac{1}{2} - \gamma_t, \gamma_t > 0$.

a. What is the probability that the classifier h_1 (selected at round 1) will be selected again at round 2? Justify your answer. **(0.75 points)**

b. Consider $\gamma = min\{\gamma_1, \gamma_2, \gamma_3\}$. Show that the training error of the final classifier h_{final} is at most $\frac{1}{2} - \frac{3}{2}\gamma + \gamma^2$ and show that this is strictly smaller than $\frac{1}{2} - \gamma$. **(0.75 points)**

Solution

a. We will prove that under the constraint $\epsilon_i <= 1/2 - \gamma_i, \gamma_i > 0, i = \overline{1,3}$, there is 0 probability of choosing h2 = h1 in the second round.

From Seminar 6, we have the following relationship from $\mathcal{D}_{t+1}$ in relation to $\mathcal{D}_t$:

$\mathcal{D}_{t+1}(i) = \frac{\mathcal{D}_t(i) * e^{-w_t * h_t(x_i) * y_i}}{Z_{t+1}}$.
Z_{t+1} is the normalizing factor.

$w_t = \frac{1}{2} * \log(\frac{1}{\epsilon_t} - 1)$ is the weight

$\epsilon_t = \mathcal{P}_{i \sim D_t}[h_t(x) \neq y_i] = \sum_{h_t(x_i) \neq y_i} \mathcal{D}_t(i)$.

From Seminar 6 on AdaBoost, we know that :

- If example x_i is correctly classified, then $h_t(x_i) = y_i$, so at the next iteration $t + 1$, its importance (probability distribution) is decreased to $D_{(t+1)}(i) = \frac{D_{(t)}(i) * \sqrt{\frac{\epsilon_t}{1-\epsilon_t}}}{\mathcal{Z}_{t+1}}$
- If example x_i is misclassified, then $h_t(x_i) = y_i$, so at the next iteration $t + 1$, its importance (probability distribution) will be increased to $D_{(t+1)}(i) = \frac{D_{(t)}(i) * \sqrt{\frac{1-\epsilon_t}{\epsilon_t}}}{\mathcal{Z}_{t+1}}$

Then, $Z_{t+1} = \sum_{h_t(x_i) = y_i} \mathcal{D}_t(i) \sqrt{\frac{\epsilon_t}{1-\epsilon_t}} + \sum_{h_t(x_i) \neq y_i} \mathcal{D}_t(i) \sqrt{\frac{1-\epsilon_t}{\epsilon_t}}$.

$Z_{t+1} = (1 - \epsilon_t) * \sqrt{\frac{\epsilon_t}{1-\epsilon_t}} + \epsilon_t * \sqrt{\frac{\epsilon_t}{1-\epsilon_t}} = 2 * \sqrt{\epsilon_t * (1 - \epsilon_t)}$. (*)

If the classifier h_1 would be selected again at round 2, then $h_2 = h_1$.

- $Z_2 = 2 * \sqrt{\epsilon_1 * (1 - \epsilon_1)}$ from (*)
- $D_2(i) = \frac{D_1(i) * \sqrt{\frac{1-\epsilon_1}{\epsilon_1}}}{Z_2}$, if $h_1(x_1) \neq y_1$
- $\epsilon_2 = \sum_{h_2(x_i) \neq y_i} D_2(i) = \sum_{h_1(x_i) \neq y_i} \frac{D_1(i) * \sqrt{\frac{1-\epsilon_1}{\epsilon_1}}}{Z_2} = \sum_{h_1(x_i) \neq y_i} * D_1(i) * \frac{\sqrt{\frac{1-\epsilon_1}{\epsilon_1}}}{Z_2} = \epsilon_1 * \frac{\sqrt{\frac{1-\epsilon_1}{\epsilon_1}}}{Z_2}$

So, $\epsilon_2 = \epsilon_1 * \frac{\sqrt{\frac{1-\epsilon_1}{\epsilon_1}}}{2 * \sqrt{\epsilon_1 * (1-\epsilon_1)}} = \epsilon_1 * \sqrt{\frac{1-\epsilon_1}{\epsilon_1}} * \frac{1}{2 * \sqrt{\epsilon_1 * (1-\epsilon_1)}} = \epsilon_1 * \frac{1}{\sqrt{\epsilon_1}} * \frac{1}{2 * \sqrt{\epsilon_1}} = \frac{1}{2}$.

But, in the assumption above, we had that every $\epsilon_i <= 1/2 - \gamma_i, \gamma_i > 0, i = \overline{1,3}$, thus $\epsilon_2 <= 1/2 - \gamma_2$. So, $1/2 <= 1/2 - \gamma_2 \rightarrow \gamma_2 >= 0$.

Given that, in the same assumption above, we have $\gamma_2 < 0$, we have a contradiction.

4. **(1 point)** 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 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 learning 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}.

Hint: Find an algorithm that returns $h(x) = 0$ or the disjunction of 2 literals.

Solution

A learning algorithm $\mathcal{A}$ is a γ-weak-learner for $\mathcal{H}$ if there exists a function $m_{\mathcal{H}} : (0,1) \to \mathbf{N}$:

- for every $\delta > 0$ (confidence),
- for every labeling function $f \in \mathcal{H}$, $f : \mathcal{X} \to \{-1, +1\}$ (realizability case)
- for every distribution $\mathcal{D}$ over $\mathcal{X}$,

when we run the algorithm $\mathcal{A}$ 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 returns a hypothesis h (h might not be from $\mathcal{H}$, in the case of improper learning, such that, with probability at least $1 - \delta$, over the choice of examples, $L_{D,f} \leq \frac{1}{2} - \gamma$.

Using the distribution rule, we can transform the 2-term disjunctive normal form to a 2-term conjunctive normal form formula:

$A_1 \vee A_2 = \bigwedge(a_1 \vee a_2) = \bigwedge y_{a_1,a_2}$, where $a_1 \in A_1$ and $a_2 \in A_2$.

By doing this, we obtain a conjunction of $(2n)^2$ variables. Each of them is a disjunction of 2 literals from the original conjunctions.

We know that C_N, the concept class of conjunctions of at most n boolean literals is PAC learnable with the sample complexity $m_{\mathcal{H}}(\epsilon, \delta) = [\frac{1}{\epsilon}(n \log(3) - \log(\delta)]$.

So our conjunction is also PAC learnable, but with $m_{\mathcal{H}}(\epsilon, \delta) = [\frac{1}{\epsilon}((2 * n)^2 \log(3) - \log(\delta)]$

Ex-oficio: 0.5 points.