

CS 228: Computational Learning Theory
Homework 4 (Take Home Exam) Problems
Mar. 28th - Apr. 4th

Policy reminders: You are strongly encouraged to type your solutions using LATEX. You may *not* discuss this problem set with any one except the course staff. You are not allowed to use any materials except your own class notes, the course textbooks, and any material posted on the class website or handed out in class. Moreover, you are not allowed to query any LLM regarding any part of any problem. This problem set is due on **April 4**; *no late days are allowed*. For any clarifications or questions, please send an email to the teaching staff.

Problem 1 (10 pts). Few Relevant Variables Implies Efficient PAC Learnability.

A variable x_i is said to be *relevant* to a boolean function f if there are two truth assignments A, B which differ only in their assignment to x_i but which have $f(A) \neq f(B)$. Let \mathcal{C} be the class of boolean functions over $\{0, 1\}^n$ that have at most $\log n$ relevant variables. Give a direct proof (i.e., without reducing to other classes) that \mathcal{C} is PAC-learnable in polynomial time by membership queries.

Problem Setup: Let $f : \{0, 1\}^n \rightarrow \{0, 1\}$ be a Boolean function depending on at most $\log n$ variables. We give a direct polynomial-time learning algorithm using membership queries:

Identify relevant variables: For each $i \in 1, \dots, n$, we test whether x_i is relevant to f . Fix some reference assignment $a \in \{0, 1\}^n$ (for example, $a = (0, 0, \dots, 0)$) and obtain $f(a)$ via a membership query. Then for each i , flip the i -th bit of a to get $a^{(i)}$ (so $a_j^{(i)} = a_j$ for $j \neq i$ and $a_i^{(i)} = 1 - a_i$) and query $f(a^{(i)})$. If $f(a^{(i)}) \neq f(a)$, then x_i is relevant; if $f(a^{(i)}) = f(a)$, then x_i is probably irrelevant. Since f has at most $\log n$ relevant variables, most coordinates will appear irrelevant under any single assignment. If a truly relevant variable x_i happens not to change f when flipped (i.e. $f(a^{(i)}) = f(a)$ despite x_i being relevant), it means that the particular assignment to the other relevant variables in a caused f to be insensitive to x_i . In that case, we can repeat the test under a different assignment. Because there are at most $\log n$ relevant variables, a random choice of a will, with high probability, assign the other relevant bits in a way that reveals x_i 's influence. In fact, by trying at most $O(\log n)$ different random assignments as bases, we can ensure that every relevant coordinate flips the output in at least one trial. Thus, we can find the exact set $R \subseteq 1, \dots, n$ of relevant indices using at most $O(n \log n)$ queries.

Learn f on relevant coordinates: Let $r = |R| \leq \log n$. Now f is effectively a function of the r bits in R . We can determine f exactly by querying all 2^r possible assignments to those r bits. For each $y \in \{0, 1\}^R$ (an assignment to the bits in R), construct any $x \in \{0, 1\}^n$ such that $x_R = y$ (bits in R set according to y) and $x_j = 0$ for $j \notin R$ (arbitrary values work for irrelevant bits). Query $f(x)$ to obtain the value $f(y)$ (since bits outside R do not matter). Collecting these results for all $2^r \leq 2^{\log n} = n$ assignments, we now have the truth-table of f as a function of its relevant inputs.

Conclusion: Finally, output any hypothesis h that depends only on R and matches this truth-table (for example, a DNF or decision tree consistent with those 2^r values). By construction, $h(x) = f(x)$

for every $x \in \{0, 1\}^n$. The algorithm uses at most $n \log n + n = O(n \log n)$ membership queries and runs in $\text{poly}(n)$ time. Therefore, \mathcal{C} is PAC-learnable (in fact, exactly learnable) in polynomial time with membership queries.

Problem 2 (15 pts). **Boolean Threshold Functions Are Not PAC-learnable.** Given any $y \in \{0, 1\}^n$ and any integer $k \geq 0$, the boolean threshold function $TH_{k,y}$ is defined as follows: an input $x \in \{0, 1\}^n$ is a positive example iff $x \cdot y \geq k$, where $x \cdot y$ denotes the standard real-valued dot product of two n -dimensional vectors. Prove that if $\text{NP} \neq \text{RP}$ then the representation class of boolean threshold functions is not PAC learnable by boolean threshold functions.

Hint. Reduce from the Zero-One Integer Programming problem (ZIP) which is known to be NP-complete. An instance of ZIP is a set of s pairs $\langle c_i, b_i \rangle$ and the pair $\langle \bar{a}, B \rangle$, where $c_i \in \{0, 1\}^n$, $\bar{a} \in \{0, 1\}^n$, $b_i \in \{0, 1\}$, and $0 \leq B \leq n$. The problem is to determine whether there exists a vector $\bar{d} \in \{0, 1\}^n$ such that $c_i \cdot \bar{d} \leq b_i$ for $1 \leq i \leq s$ and $\bar{a} \cdot \bar{d} \geq B$.

Proof by reduction from ZIP: Assume, for sake of contradiction, that there is a PAC learning algorithm \mathcal{A} that can properly learn the class of Boolean threshold functions in polynomial time (with membership or random examples). We use \mathcal{A} to decide the Zero-One Integer Programming problem, which is NP-complete.

An instance of ZIP consists of vectors $c_1, \dots, c_s, \bar{a} \in \{0, 1\}^n$, integers $b_1, \dots, b_s \in \{0, 1\}$ and B (with $0 \leq B \leq n$). We need to determine if there exists $d \in \{0, 1\}^n$ such that for all $1 \leq i \leq s$, $c_i \cdot d \leq b_i$, and $\bar{a} \cdot d \geq B$. Construct the following labeled sample S for a learning problem:

- For each constraint i with $b_i = 0$, include the example $x = c_i$ with label $\ell(x) = 0$.
- For each constraint i with $b_i = 1$, include, for every distinct pair of 1-coordinates (p, q) in c_i , an example $x_{p,q}^{(i)}$ that has $x_{p,q}^{(i)}(p) = x_{p,q}^{(i)}(q) = 1$ and zeros in all other coordinates, with label $\ell(x_{p,q}^{(i)}) = 0$.
- Include the example $x = \bar{a}$ with label $\ell(x) = 1$.

Intuitively, negative examples encode the constraints (violation should yield output 1) and the single positive example encodes the $\bar{a} \cdot d \geq B$ requirement. Now we claim:

1. If a satisfying assignment d exists for the ZIP instance, then there is a threshold function $h(y) = 1 \iff y \cdot d \geq 1$ that correctly classifies all examples in S . Specifically, set the weight vector to d and threshold $k = 1$. For each original constraint i :
 - If $b_i = 0$, d satisfies $c_i \cdot d \leq 0$, hence $c_i \cdot d = 0$ and $h(c_i) = 0$ (correct label).
 - If $b_i = 1$, d satisfies $c_i \cdot d \leq 1$. Then for any pair (p, q) of ones in c_i , at most one of p, q lies in d . Thus $x_{p,q}^{(i)} \cdot d \leq 1$, so $h(x_{p,q}^{(i)}) = 0$ (correct).
 - $\bar{a} \cdot d \geq B \geq 1$, so $h(\bar{a}) = 1$ (correct). Therefore h (a threshold function in our class) perfectly fits all examples in S .
2. If no satisfying d exists, then no threshold function can perfectly classify S . In particular, consider any threshold hypothesis $h_{k,y}(x) = \mathbf{1}_{y \cdot x \geq k}$. We examine its performance on the sample:
 - If h outputs 1 for \bar{a} (to get \bar{a} 's label correct), then $y \cdot \bar{a} \geq k$. Since \bar{a} has at most n ones, $k \leq n$. For each constraint i , because d doesn't exist, any y must violate at least one constraint. Suppose y violates constraint j , meaning $c_j \cdot y \geq b_j + 1$ (if $b_j = 0$, then

$c_j \cdot y \geq 1$; if $b_j = 1$, then $c_j \cdot y \geq 2$). Then one of the negative examples derived from c_j will satisfy $y \cdot x \geq k$ and be labeled 1 by h , causing an error. For example, if $b_j = 0$ and $c_j \cdot y \geq 1$, then $h(c_j) = 1$ (since $y \cdot c_j \geq 1 \geq k$) but $\ell(c_j) = 0$. If $b_j = 1$ and $c_j \cdot y \geq 2$, pick two coordinates p, q where y has ones (contributing to the sum); then $y \cdot x_{p,q}^{(j)} \geq 2 \geq k$, so $h(x_{p,q}^{(j)}) = 1$ but $\ell(x_{p,q}^{(j)}) = 0$.

- If h outputs 0 for \bar{a} (to avoid violating some constraint), then $\bar{a} \cdot y < k$. But $\ell(\bar{a}) = 1$, so h fails on \bar{a} .

In either case h makes an error on S . In fact, any threshold hypothesis must err on at least one example when no solution exists.

Consequently, if \mathcal{A} could learn threshold functions, we would run \mathcal{A} on the sample S . With high probability, it would output a hypothesis h . We then simulate h on the examples in S to check its accuracy. If h labels all examples correctly (zero errors), we conclude a satisfying d exists (and indeed y from h is a valid solution for ZIP). If h has any error on S , we conclude no solution exists. This procedure runs in randomized polynomial time, deciding the NP-complete problem ZIP. Therefore, $\text{NP} = \text{RP}$ would follow. By contrapositive, if $\text{NP} \neq \text{RP}$, no such polynomial PAC learning algorithm \mathcal{A} can exist. In other words, the class of boolean threshold functions is not PAC-learnable by (proper) threshold hypotheses under the $\text{NP} \neq \text{RP}$ assumption.

Problem 3 (15 pts). Exact Learning DNFs via Membership Queries. Prove that the class of $(\log n)$ -term DNF is learnable in polynomial time in the exact learning model using membership and equivalence queries.

Hint: Reduce this problem to that of learning a DFA (deterministic finite automaton).

We show how to learn a DNF formula with at most $\log n$ terms by reducing the problem to learning a polynomial-size DFA.

Construction of DFA: Let c be the unknown DNF, consisting of $t \leq \log n$ terms T_1, \dots, T_t (each T_j is a conjunction of literals). We construct a DFA M over the alphabet $\{0, 1\}$ that reads strings of length n (each n -bit string is an input to c) and accepts exactly those strings x for which $c(x) = 1$. The state of M after reading a prefix of the input will record which of the t terms have not yet been falsified by that prefix. Formally, define the state set

$$Q = \{S \subseteq \{1, \dots, t\} : S \text{ is the set of terms consistent with the prefix read so far}\}.$$

Initially, M starts in state $S_0 = \{1, \dots, t\}$ (all terms are alive). The transition function δ is defined as follows: if the automaton is in state $S \subseteq [t]$ and the next input bit is b in position i , then

$$\delta(S, b) = \{j \in S : \text{the } i\text{-th literal of term } T_j \text{ is consistent with bit } b\}.$$

After reading all n bits, the DFA accepts if and only if $S \neq \emptyset$, i.e., at least one term is satisfied. Since there are at most $\log n$ terms, the number of subsets S is at most $2^{\log n} = n$, so M has at most n states.

Learning via Angluin's L* algorithm: We can simulate the exact learning of M using Angluin's algorithm, which requires membership and equivalence queries. A membership query to M can be answered by querying c directly. An equivalence query for a hypothesized DFA can also be answered by querying c for agreement. Since M has $O(n)$ states, the DFA learning algorithm runs in polynomial time. Once M is learned, it can be converted back into a DNF with at most $\log n$ terms. Thus, $(\log n)$ -term DNF is exactly learnable in polynomial time with membership and equivalence queries.

Problem 4. (16pt) **Teaching dimension.** Let X be a finite instance space. Given a concept class \mathcal{C} and a target concept $c \in \mathcal{C}$, we say that a sequence T of labelled examples is a *teaching sequence* for c in \mathcal{C} if c is the only concept in \mathcal{C} that is consistent with T . Let $T(c)$ be the set of all teaching sequences for c in \mathcal{C} . The *teaching dimension* of concept class \mathcal{C} is then defined to be

$$TD(\mathcal{C}) = \max_{c \in \mathcal{C}} \min_{T \in T(c)} |T|$$

where $|T|$ denotes the number of examples in the sequence T .

1. (4pt) Give an example of a concept class \mathcal{C} for which $TD(\mathcal{C}) > VC\text{-dim}(\mathcal{C})$.
2. (4pt) Give an example of a concept class \mathcal{C} for which $TD(\mathcal{C}) < VC\text{-dim}(\mathcal{C})$.
3. (4pt) Show that for any concept class \mathcal{C} , $TD(\mathcal{C}) \leq |\mathcal{C}| - 1$.
4. (4pt) Show that for any concept class \mathcal{C} , $TD(\mathcal{C}) \leq VC\text{-dim}(\mathcal{C}) + |\mathcal{C}| - 2^{VC\text{-dim}(\mathcal{C})}$.

(1) Example where $TD(\mathcal{C}) > VC\text{-dim}(\mathcal{C})$: Consider the concept class \mathcal{C} defined over instance space $X = \{p, q\}$ (two points) with the following concepts:

- c_1 : $p = 1, q = 1$ (labels both points positive),
- c_2 : $p = 1, q = 0$,
- c_3 : $p = 0, q = 1$.

Here $VC\text{-dim}(\mathcal{C}) = 1$ because no set of size 2 can be shattered (for example, the labeling $p = 0, q = 0$ is not realized by any concept in \mathcal{C}). However, the teaching dimension $TD(\mathcal{C})$ is 2. To see this, note that for concept c_1 (the all-positive concept), any teaching sequence must distinguish it from both c_2 and c_3 . For instance, one optimal teaching sequence is $\{(p, 1), (q, 1)\}$: labeling both p and q as positive is only consistent with c_1 . This sequence has length 2. Likewise, c_2 can be taught with one example $(q, 0)$ (which eliminates c_3 and c_1), and c_3 can be taught with $(p, 0)$. The largest minimum teaching sequence size among c_1, c_2, c_3 is 2 (for c_1), so $TD(\mathcal{C}) = 2$. Thus $TD(\mathcal{C}) = 2 > VC\text{-dim}(\mathcal{C}) = 1$.

Another simple example: Let \mathcal{C} consist of one "all-negative" concept c_\emptyset (labels every point 0) and n "singleton" concepts c_i each labeling exactly one distinct point positive and all others negative. The VC-dimension of this class is 1 (we cannot shatter any 2 points, since no concept labels two points positive). However, the teaching dimension is n . Concept c_\emptyset requires n negative examples (one for each point) in any teaching sequence, because to eliminate each c_i the teacher must show that point i is labeled 0. For the singletons c_i , one positive example of point i suffices to eliminate all others. Thus $TD(\mathcal{C}) = n$ while $VC\text{-dim}(\mathcal{C}) = 1$.

(2) Example where $TD(\mathcal{C}) < VC\text{-dim}(\mathcal{C})$: Let \mathcal{C} be the class of threshold functions on a line of N points $\{1, 2, \dots, N\}$. For each $0 \leq k \leq N$, concept c_k labels a point x as positive iff $x \leq k$. Intuitively, c_k classifies all points up to some cutoff k as 1, and points greater than k as 0. The VC-dimension of this class is N (in fact, it shatters any set of points because we can always realize any desired labeling pattern that has a single switch from 0 to 1 at the appropriate threshold). However, the teaching dimension is 2. Any threshold function c_k can be uniquely identified by the teacher with just two examples: for instance, the teacher can present $(k, 1)$ and $(k + 1, 0)$ (the point at the threshold labeled positive and the next point labeled negative). No other threshold $c_{k'}$ for $k' \neq k$ is consistent with those two labels, so this is a valid teaching sequence of length 2. Indeed, we have $TD(\mathcal{C}) = 2$ while $VC\text{-dim}(\mathcal{C}) = N$, so $TD(\mathcal{C}) \ll VC\text{-dim}(\mathcal{C})$ for large N .

(3) Proof that $TD(\mathcal{C}) \leq |\mathcal{C}| - 1$: In the worst case, a teacher can always distinguish a target concept $c \in \mathcal{C}$ from all other $|\mathcal{C}| - 1$ concepts by providing one example per competing concept. Specifically, list the other concepts as $\mathcal{C} \setminus \{c\} = \{h_1, h_2, \dots, h_m\}$ where $m = |\mathcal{C}| - 1$. For each h_i , because $h_i \neq c$, there exists at least one instance x_i such that $c(x_i) \neq h_i(x_i)$. The teacher can include the labeled example $(x_i, c(x_i))$ in the teaching sequence. This example will be consistent with c 's labeling by definition, but it will conflict with h_i (since $h_i(x_i)$ is the opposite label). Thus h_i is eliminated. After including one such example for every other concept, only c remains consistent. The total number of examples used is $|\mathcal{C}| - 1$. This shows $\forall c, \min T(c) \leq |\mathcal{C}| - 1$, and hence $TD(\mathcal{C}) = \max_c \min T(c) \leq |\mathcal{C}| - 1$.

(4) Proof that $TD(\mathcal{C}) \leq VC\text{-dim}(\mathcal{C}) + |\mathcal{C}| - 2^{VC\text{-dim}(\mathcal{C})}$: Let $d = VC\text{-dim}(\mathcal{C})$. By definition, there exists a set S of d instances shattered by \mathcal{C} . That means every binary labeling on S is realized by some concept in \mathcal{C} , so $|\mathcal{C}| \geq 2^d$ (at least one concept per dichotomy on S). Now fix the target concept $c \in \mathcal{C}$. Among the 2^d possible labelings of S , c 's labeling (the restriction $c|_S$) is one of them.

Consider two stages for teaching c :

Stage 1: The teacher provides the d labeled examples for every point in S according to c . This uniquely specifies c 's labeling pattern on S . After this stage, any concept $h \in \mathcal{C}$ that remains consistent must have $h(x) = c(x)$ for all $x \in S$. By shattering, there is exactly one concept in \mathcal{C} for each distinct pattern on S , so at most those concepts that share c 's S -pattern survive. In the worst case, initially $|\mathcal{C}|$ concepts were possible; after stage 1, at most $|\mathcal{C}|/2^d$ concepts remain (because 2^d distinct patterns were possible on S , and all but one pattern — c 's pattern — have been eliminated for the surviving concepts).

Stage 2: Now we have a reduced concept class $\mathcal{C}' = \{h \in \mathcal{C} : h|_S = c|_S\}$ consisting of all concepts that agreed with c on S . Notice that $c \in \mathcal{C}'$. The teacher must eliminate all $h \in \mathcal{C}'$ with $h \neq c$. In the worst case, $|\mathcal{C}'|$ could be as large as $|\mathcal{C}|/2^d$. Using the same argument as in part 3, the teacher can eliminate each remaining incorrect concept with at most one additional example. So stage 2 uses at most $|\mathcal{C}'| - 1$ examples.

Counting examples: Stage 1 used d examples, and stage 2 used at most $|\mathcal{C}'| - 1$ examples. So one valid teaching sequence for c has length $\leq d + (|\mathcal{C}'| - 1)$. Therefore:

$$TD(\mathcal{C}) \leq d + (|\mathcal{C}'| - 1) \leq d + \left(\frac{|\mathcal{C}|}{2^d} - 1 \right) = d + \frac{|\mathcal{C}|}{2^d} - 1.$$

Since $|\mathcal{C}'| \leq |\mathcal{C}| - (2^d - 1)$ (at least $2^d - 1$ concepts from \mathcal{C} were eliminated in stage 1, leaving at most $|\mathcal{C}| - 2^d + 1$ survivors; using this bound yields the same inequality above), we can simplify the bound to:

$$TD(\mathcal{C}) \leq VC\text{-dim}(\mathcal{C}) + |\mathcal{C}| - 2^{VC\text{-dim}(\mathcal{C})}.$$

This is exactly the desired inequality. Intuitively, the worst-case teaching sequence length is achieved when the target concept's label pattern on a shattered set S is shared by as many concepts as possible (i.e., all other concepts differ only outside S), forcing the teacher to use the maximum number of stage-2 examples. But even in that case, stage 1 removes an exponential number of concepts (those with different S -patterns), greatly reducing the pool that must be distinguished one-by-one. Hence the teaching dimension grows more slowly than the number of concepts, incorporating the effect of VC-dimension.

Problem 5 (15 pts). Persistent Classification Noise Model. For learning from noisy examples using membership queries, the random *persistent* classification noise model is as follows: Given

concept $c \in \mathcal{C}$ and noise rate η , a noisy concept c' is produced by flipping the label of c at each point with probability η . Then a learning algorithm gets all examples and membership queries labeled according to c' . This implies that in this model, with certain (albeit negligible) probability, c' might be very far from c . For simplicity, we only require that a learning algorithm succeeds with probability at least $1/2$, where the probability of success also depends on the random choice of c' , and, as before, the running time can be polynomial in

$$\frac{1}{1 - 2\eta}.$$

Let \mathcal{P} be the class of parity functions over $\{0, 1\}^n$ and \mathcal{U} be the uniform distribution over $\{0, 1\}^n$.

- (a) (5pts) Give an algorithm that learns \mathcal{P} using membership queries in the presence of random persistent classification noise (that is, for any $\eta < 1/2$).
- (b) (5pts) Now assume that c' is chosen by an adversary such that

$$\Pr_{\mathcal{U}}[c'(x) \neq c(x)] \leq \eta.$$

Give an algorithm that learns \mathcal{P} using membership queries in the presence of such malicious noise of rate $\eta = 1/5$.

- (c) (5pts) Prove that \mathcal{P} is not learnable using membership queries with the malicious noise as above of rate $\geq 1/4$.

Let \mathcal{P} be the class of parity functions on n boolean variables, and let the target parity be $c(x) = x \cdot w \pmod{2}$ for some unknown $w \in \{0, 1\}^n$. We consider learning c using membership queries under two noise models:

(a) Random persistent classification noise (any $\eta < 1/2$): In this model, a single “noisy” target function c' is fixed at the start by flipping each output bit of c independently with probability η . Equivalently, there is some unknown error function $e : \{0, 1\}^n \rightarrow \{0, 1\}$ such that for each x , $c'(x) = c(x) \oplus e(x)$, where $\Pr_{x \sim \mathcal{U}}[e(x) = 1] = \eta$ (and these flips are random and independent across x). The learner has membership query access to c' (not c directly).

Learning algorithm: We can exploit the linear structure of parity and use pairwise queries to cancel out noise. The core idea is to query c' on two carefully chosen inputs and combine the answers (via XOR) to obtain information about w with a lower noise rate. For any two inputs $x, y \in \{0, 1\}^n$, note that

$$c'(x) \oplus c'(y) = (c(x) \oplus e(x)) \oplus (c(y) \oplus e(y)) = c(x \oplus y) \oplus (e(x) \oplus e(y)).$$

If we choose x and y such that $x \oplus y = e_i$ (the unit vector with 1 in the i -th coordinate and 0 elsewhere), then $c(x \oplus y) = w_i$. Thus,

$$c'(x) \oplus c'(y) = w_i \oplus (e(x) \oplus e(y)).$$

The XOR of two independent $\text{Ber}(\eta)$ bits is 1 with probability $2\eta(1 - \eta) < 1/2$ (since $\eta < 1/2$). Therefore, each such pair-query gives a noisy version of w_i with error rate $\eta' = 2\eta(1 - \eta) < 1/2$.

Repeat $O(\log n)$ times per coordinate, taking a majority vote to recover w_i with high probability. Doing this for all n coordinates allows us to recover the entire w . Total queries: $O(n \log n)$.

(b) Adversarial noise of rate $\eta = 1/5$: Now c' is an arbitrary function (not necessarily a parity) that differs from c on at most 20

Learning algorithm: For each $i = 1, \dots, n$, choose $x \sim \{0, 1\}^n$ uniformly at random, and compute $c'(x) \oplus c'(x \oplus e_i)$. With probability ≥ 0.6 , either both are correct or both are flipped, so the XOR yields w_i correctly. With probability ≤ 0.4 , exactly one value is flipped, producing the wrong w_i .

Repeating $O(\log n)$ times and taking a majority vote gives correct w_i with high probability. Repeating for all i , we reconstruct w using $O(n \log n)$ queries. Thus parity is learnable under adversarial noise rate $< 1/4$.

(c) Impossibility for $\eta \geq 1/4$: We show that for $\eta \geq 1/4$, no algorithm making $\text{poly}(n)$ membership queries can learn parity functions.

Let $c(x) = x \cdot w$ be the target parity. Let the learner make at most $q = \text{poly}(n)$ queries. Choose another parity function \tilde{c} that differs from c on exactly half the domain (e.g., $w' = w \oplus e_1$).

Partition the domain into:

- $A = \{x : c(x) = \tilde{c}(x)\}$ (size 2^{n-1}),
- $B = \{x : c(x) \neq \tilde{c}(x)\}$ (size 2^{n-1}).

Define c' as:

$$c'(x) = \begin{cases} c(x) & x \notin B' \\ \tilde{c}(x) & x \in B' \end{cases}$$

for $B' \subseteq B$ with $|B'| = 2^{n-2}$ (i.e., 25% of the domain). Choose B' so that $Q \cap B' = \emptyset$, where Q is the set of queries made by the learner. Then all queries return values consistent with both c and c' . But c and c' differ on 25% of the domain, so the learner cannot output a hypothesis with error $< 1/4$ with high confidence.

Hence, parity is not learnable under adversarial noise with $\eta \geq 1/4$.