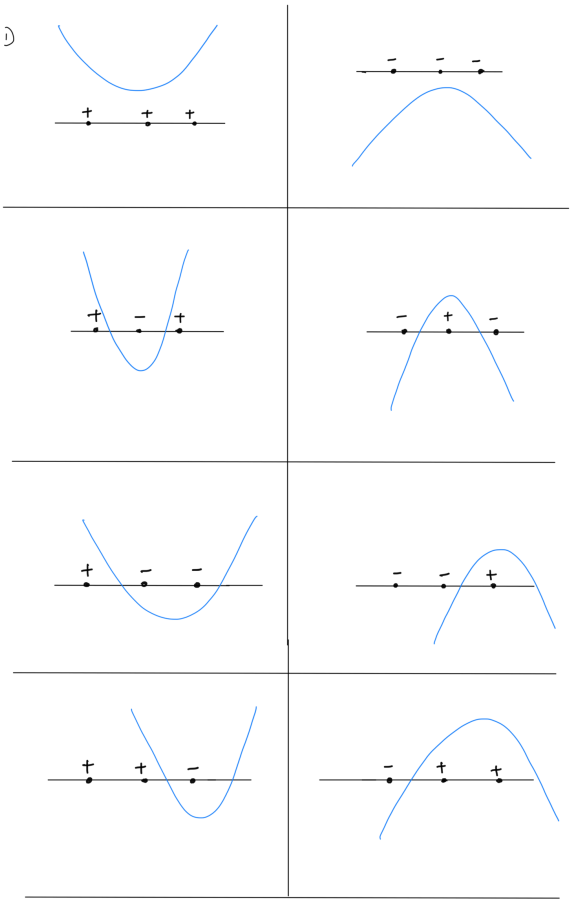
HOMEWORK 7

>>Abhay Kumar<<
>>9081403157<<

Instructions: Although this is a programming homework, you only need to hand in a pdf answer file. There is no need to submit the latex source or any code. You can choose any programming language, as long as you implement the algorithm from scratch.
Use this latex file as a template to develop your homework. Submit your homework on time as a single pdf file to Canvas. Please check Piazza for updates about the homework.

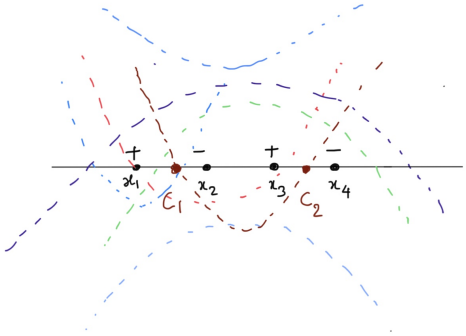
1 VC dimension (30 pts)

Let the input $x \in X = \mathbb{R}$. Consider $F = \{f(x) = \text{sgn}(ax^2 + bx + c) : a, b, c \in \mathbb{R}\}$, where $\text{sgn}(z) = 1$ if $z \geq 0$, and 0 otherwise. What is $VC(F)$? Prove it.
$VC(F) = 3$
To prove that the $VC(F) = 3$, we need to show that the there exists configurations of three points such that all possible label combinations (i.e. +++, —, +-+, -+-, +–, –+, ++-, -++) can be shattered. This shows that $VC(F) \geq 3$.



In the above figures, I assume $\text{sgn}(ax^2 + bx + c) > 0 \implies$ label = 1 else label = 0. (considering horizontal black line as y = 0)

Now, we need to show that given configuration of 4 points can't be shattered by F. From the attached figure below, we can see that any possible quadratic polynomial crosses the x-axis (or y=0) line at maximum two points, so we can accommodate at most two flips (i.e. +-+, -+-) of labels. However for 3 flips of labels (i.e +-+- or -+-+), we can't shatter using a single quadratic function as quadratic functions can change signs at most two times. This shows $VC(F) < 4$.



Lets assume that we have 4 points x_1, x_2, x_3, x_4 and lets say $x_1 \leq x_2 \leq x_3 \leq x_4$ (without any loss of generality). A quadratic function can cut x-axis at most twice (lets say at c_1 and c_2.) then for all possible placements of c_1 and c_2, it could never correctly classify all 4 points and one point will at least be wrongly labeled.
for example, assume $x_1 \leq c_1 \leq x_2 \leq c_2 \leq x_3 \leq x_4 \implies x_1, x_2, x_3$ are correctly classified and x_4 is wrongly classified (predicted =1, actual =0)
for example, assume $x_1 \leq x_2 \leq c_1 \leq x_3 \leq c_2 \leq x_4 \implies x_2, x_3, x_4$ are correctly classified and x_1 is wrongly classified (predicted =0, actual =1)

Combining arguments from above, we can conclude that $VC(F) \geq 3$ and $VC(F) < 4$, i.e $VC(F) = 3$.

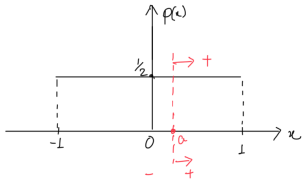
2 Verify PAC Bound (30 pts)

Let h be the VC dimension of function family F. For any $\delta > 0$, with probability at least $1 - \delta$ we have

$$R(\hat{f}_S) - \hat{R}_S(\hat{f}_S) \leq 2\sqrt{2\frac{h \log n + h \log \frac{2e}{h} + \log \frac{2}{\delta}}{n}},$$

where $R(f) = \mathbb{E}1_{[f(x) \neq y]}$ is the risk of f, $S = (x_1, y_1), \ldots, (x_n, y_n)$ is a training set of size n, $\hat{R}_S(f) = \frac{1}{n}\sum_{i=1}^{n} 1_{[f(x_i) \neq y_i]}$ is the empirical risk of f on S, and $\hat{f}_S \in \arg\min_{f \in F} \hat{R}_S(f)$ is an empirical risk minimizer (ERM).
We now verify this bound on a simple classification task. Let $p(x) = \text{uniform}([-1, 1])$. Given x, the label is deterministic: $y = \text{sgn}(x)$. Recall $\text{sgn}(x) = 1$ if $x \geq 0$, and 0 otherwise. This means the true decision boundary is at $x = 0$. Let $f_\theta(x) := \text{sgn}(x - \theta)$ which has threshold at θ. Let $F = \{f_\theta : \theta \in [-1, 1]\}$.

1. Given S, find the smallest positive item: $a = \min_{(x_i, y_i) \in S : y_i = 1} x_i$ if one exists, otherwise let $a = 1$. Is $\hat{f}_S := f_a$ an ERM? Justify your answer.
   yes, $\hat{f}_S := f_a$ is an ERM.
   for any hypothesis function, training error ≥ 0. For f_a, we get training error as 0 (see section 2.3) and hence it minimizes the empirical risk, $\hat{R}_S(f_a) = 0$, f_a minimizes the empirical risk on S to 0 and hence $f_a \in \arg\min_{f \in F} \hat{R}_S(f)$, i,e f_a is empirical risk minimizer.
2. Derive $R(f_a)$.



Given x, the label is deterministic: $y = \text{sgn}(x)$.
smallest positive item: $a = \min_{(x_i,y_i)\in S:y_i=1} x_i$ if one exists, otherwise let $a = 1$.
from the above figure, we can see that only the $\mathbf{x} \in [0, a)$ will be incorrectly classified by f_a.

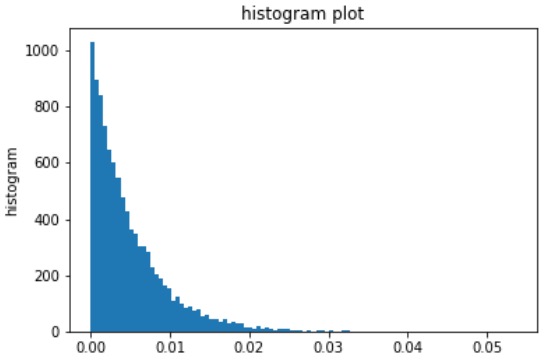
$R(f_a) = \mathbb{E}1_{[f_a(x)\neq y]} = \int_{-1}^{1} \frac{1}{2} * 1_{[f_a(x)\neq y]} dx = \int_0^a \frac{1}{2} dx = \frac{a}{2}$

3. Derive $\hat{R}_S(f_a)$.
$\hat{R}_S(f_a) = \frac{1}{n}\sum_{i=1}^{n} 1_{[f_a(x_i)\neq y_i]}$
let there are n_1 training points in $\mathbf{x} \in [-1, 0)$, n_2 training points in $\mathbf{x} \in [0, a)$ and n_3 training points in $\mathbf{x} \in [a, 1]$
$\hat{R}_S(f_a) = \frac{1}{n}[\sum_{i=1}^{n_1} 1_{[f_a(x_i)\neq 0]} + \sum_{i=1}^{n_2} 1_{[f_a(x_i)\neq 1]} + \sum_{i=1}^{n_3} 1_{[f_a(x_i)\neq 1]}]$
1^{st} term = 0 (as f_a will classify all points $< a$ as 0 label)
2^{nd} term = 0 as there is no such point, $m_2 = 0$ (using smallest positive item: $a = \min_{(x_i,y_i)\in S:y_i=1} x_i$)
3^{rd} term = 0 (as f_a will classify all points $\geq a$ as 1 label)
$\hat{R}_S(f_a) = 0$

4. What is the VC dimension of F? Note F contains threshold classifiers of the type "left negative, right positive."
since we are restricted to threshold classifiers of the type "left negative, right positive.", only configuration with negative to the left and postive to the right could be shattered by this classifier. we can't shatter the configuration $(+|-)$ of two points using the given classifier, although we could shatter $+|+, -|-, -|+$ configurations. This makes VC dimension of $F < 2$ However, all labeling configurations of 1 points can be shattered, therefore VC dimension of F = 1

5. Fix $\delta = 0.05$ (95% confidence) and $n = 200$. Compute $2\sqrt{2\frac{h\log n + h\log\frac{2e}{h} + \log\frac{2}{\delta}}{n}}$. This is natural log.
0.6536

6. Generate 10,000 random training sets from $p(x)$ and the associated labels. Each training set S has $n = 200$ points. On each S you will compute $R(f_a) - \hat{R}_S(f_a)$. Now you have 10,000 numbers. (1) Produce a histogram of them.



(2) Find the 95% quantile of them.
0.01488647

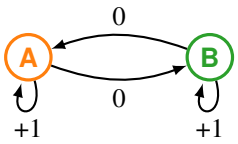
(3) Compare the 95% quantile to the bound in the previous question. Discuss your observations.
The bound calculated in Q2.5 is the worst case bound and hence it's not a tight bound.

$sup_{g\in G}(R(f_a) - \hat{R}_S(f_a))$ should be bounded by the value in Q2.5. This means the worst error difference for any ERM should be bounded by it, thereby we could achieve tighter bound for some other ERM. (Additionally, (Empirically) I observed that the error gap $(R(f_a) - \hat{R}_S(f_a))$ increases with decreasing n values and hence gets tighter bound.)

3 Q-learning (40 pts)

Consider the following Markov Decision Process. It has two states s. It has two actions a: move and stay. The state transition is deterministic: "move" moves to the other state, while "stay' stays at the current state. The reward

r is 0 for move, 1 for stay. There is a discounting factor $\gamma = 0.9$.



The reinforcement learning agent performs Q-learning. Recall the Q table has entries $Q(s, a)$. The Q table is initialized with all zeros. The agent starts in state $s_1 = A$. In any state s_t, the agent chooses the action a_t according to a behavior policy $a_t = \pi_B(s_t)$. Upon experiencing the next state and reward s_{t+1}, r_t the update is:

$$Q(s_t, a_t) \Leftarrow (1 - \alpha)Q(s_t, a_t) + \alpha \left(r_t + \gamma \max_{a'} Q(s_{t+1}, a') \right).$$

Let the step size parameter $\alpha = 0.5$.

1. Run Q-learning for 200 steps with a uniformly random behavior policy: $\pi_B(s_t)$ = move or stay with 1/2 probability for any s_t. Show the Q table at the end.

	move	stay
state A	8.47358857	9.23064391
state B	8.27699814	9.43438373

2. Reset and repeat the above, but with an ϵ-greedy behavior policy: at each state s_t, with probability $1 - \epsilon$ choose what the current Q table says is the best action: $\arg\max_a Q(s_t, a)$; Break ties arbitrarily. Otherwise (with probability ϵ) uniformly chooses between move and stay. Use $\epsilon = 0.5$.

	move	stay
state A	8.87491817	9.35634351
state B	8.36362713	9.88466698

3. Reset and repeat the above, but with a deterministic greedy behavior policy: at each state s_t use the best action $a_t \in \arg\max_a Q(s_t, a)$ indicated by the current Q table. If there is a tie, prefer move.

	move	stay
state A	0	0
state B	0	0

4. Without doing simulation, use Bellman equation to derive the true Q table induced by the MDP.

solution Approach 1:

using Bellman optimality equation for Q^*
$Q^*(s, a) = \sum_{s' \in \mathbb{S}} P(s'|s, a)[R(s, a, s') + \gamma max_{a'} Q^*(s', a')]$ – (1)
Considering all possibilities for $max_{a'} Q^*(s', a')$, we get 4 conditions/cases:

Case-1) $Q^*(A, stay) > Q^*(A, move)$ and $Q^*(B, stay) > Q^*(B, move)$
from eq (1), we get-
$Q^*(A, stay) = 1 + \gamma * Q^*(A, stay)$
$Q^*(A, move) = \gamma * Q^*(B, stay)$
$Q^*(B, stay) = 1 + \gamma * Q^*(B, stay)$
$Q^*(B, move) = \gamma * Q^*(A, stay)$

solving above equations, we get Q table as-

	move	**stay**
state A	**9**	**10**
state B	**9**	**10**

Case-2) $Q^*(A, stay) < Q^*(A, move)$ and $Q^*(B, stay) > Q^*(B, move)$
$Q^*(A, stay) = 1 + \gamma * Q^*(A, move)$
$Q^*(A, move) = \gamma * Q^*(B, stay)$
$Q^*(B, stay) = 1 + \gamma * Q^*(B, stay)$

$Q^*(B, move) = \gamma * Q^*(A, move)$

solving above equations, we get Q table as-

	move	stay
state A	9	9.1
state B	8.1	10

we **reject the above solution as it invalidates the assumption** $Q^*(A, stay) < Q^*(A, move)$

Case-3) $Q^*(A, stay) > Q^*(A, move)$ and $Q^*(B, stay) < Q^*(B, move)$
$Q^*(A, stay) = 1 + \gamma * Q^*(A, stay)$
$Q^*(A, move) = \gamma * Q^*(B, move)$
$Q^*(B, stay) = 1 + \gamma * Q^*(B, move)$
$Q^*(B, move) = \gamma * Q^*(A, stay)$

solving above equations, we get Q table as-

	move	stay
state A	8.1	10
state B	9	9.1

we **reject the above solution as it invalidates the assumption** $Q^*(B, stay) < Q^*(B, move)$

Case-4) $Q^*(A, stay) < Q^*(A, move)$ and $Q^*(B, stay) < Q^*(B, move)$
$Q^*(A, stay) = 1 + \gamma * Q^*(A, move)$
$Q^*(A, move) = \gamma * Q^*(B, move)$
$Q^*(B, stay) = 1 + \gamma * Q^*(B, move)$
$Q^*(B, move) = \gamma * Q^*(A, move)$

solving above equations, we get Q table as-

	move	stay
state A	0	1
state B	0	1

we **reject the above solution as it invalidates the assumption** $Q^*(A, stay) < Q^*(A, move)$

From all above 4 cases, we get only one valid case and the final Q table as-

	move	stay
state A	9	10
state B	9	10

solution Approach 2:
using Bellman Equation of the Q Action-Value function:-
$Q^\pi(s, a) = \sum_{s' \in \mathbb{S}} P(s'|s, a)[R(s, a, s') + \gamma \sum_{a'} \pi(a'|s') Q^\pi(s', a')]$
For sake of iterating state-action pair infinitely, let us consider the strategy to alternate between stay and move (i.e. Stay-Move-Stay-Move strategy.....). For this strategy, we observe the following update equations-

$Q^*(A, stay) = 1 + \gamma * max_{a'} Q^*(s', a')$
$Q^*(A, move) = \gamma * max_{a'} Q^*(s', a')$
$Q^*(B, stay) = 1 + \gamma * max_{a'} Q^*(s', a')$
$Q^*(B, move) = \gamma * max_{a'} Q^*(s', a')$
These update equations start to converge and form a geometric progression and could be approximated by GP infinite sum-
$Q^*(A, stay) = 1 + \gamma * (1 + \gamma + \gamma^2 + ...) = 1 + \frac{1}{1-\gamma} = 10$
$Q^*(A, move) = \gamma * (1 + \gamma + \gamma^2 + ...) = \frac{\gamma}{1-\gamma} = 9$
$Q^*(B, stay) = 1 + \gamma * (1 + \gamma + \gamma^2 + ...) = 1 + \frac{1}{1-\gamma} = 10$
$Q^*(B, move) = \gamma * (1 + \gamma + \gamma^2 + ...) = \frac{\gamma}{1-\gamma} = 9$