CIS 4190/5190 Final Exam

Version A

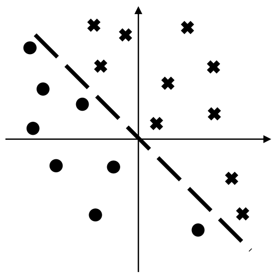
December 22, 2022

Instructions

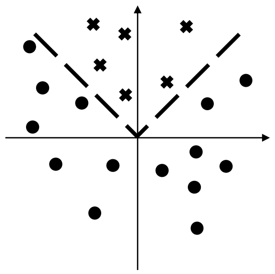
- Write your answers on paper with a pen. **Write your name, section number (4190 or 5190), and exam version (shown above)** prominently on the first page of your answers at the top left.
- No devices or cheat sheet(s) are allowed.
- The exam contains 11 questions, with 80 points total. Questions 1-7 are short answer, and 8-11 are more involved.
- Each point should take approximately 1-2 minutes; if you find yourself spending too much time on one problem, move on and come back to it.
- At the end of 2 hours, you will put down your pens and submit your exam.

Good luck!

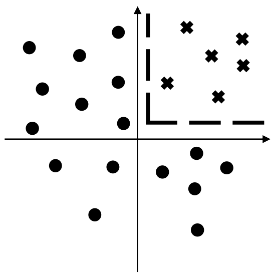
1. (12 pts) Consider the following 2D binary classification datasets:



(A)



(B)



(C)

For each of the following model families, indicate which of the above datasets can be perfectly classified by some model in the model family.

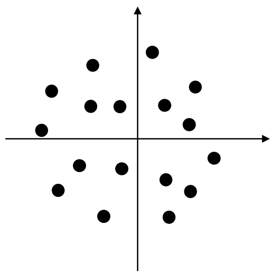
(a) (3 pts) Logistic regression A

(b) (3 pts) Logistic regression over features $\phi(x) = \begin{bmatrix} 1 & x_1 & |x_1| & x_2 \end{bmatrix}^\top$ A, B

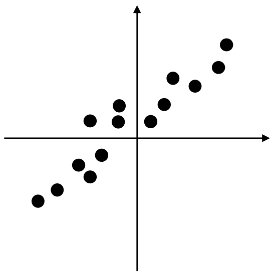
(c) (3 pts) A decision tree with axis aligned splits—i.e., $x_i \leq t$, where $i \in \{1, 2\}$ is a feature index and $t \in \mathbb{R}$ is a real-valued threshold. C

(d) (3 pts) Decision tree has oblique splits—i.e., $a_1 x_1 + a_2 x_2 \leq t$, for some $a_1, a_2, t \in \mathbb{R}$. A, B, C

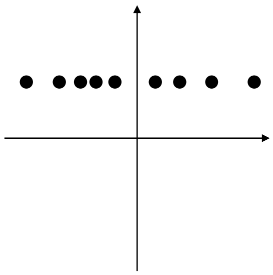
2. (4 pts) Consider the following 2D datasets:



(A)



(B)



(C)

Note that in (C), the points lie on a line. Suppose we run PCA, take only the top principal component, and use it to compress the data.

(a) (1 pt) Which dataset will have the highest reconstruction error? A

(b) (1 pts) Which dataset will have the lowest reconstruction error? C

(c) (2 pts) For your answer to part (b), what is its reconstruction error? 0

3. (4 pts) Suppose we use k-means clustering for binary classification as follows. Given a labeled dataset $\{(x_i, y_i)\}_{i=1}^n$, where $x_i \in \mathbb{R}^d$ and $y_i \in \{0, 1\}$, we first use k-means clustering to compute centroids $x^{(1)}, ..., x^{(k)} \in \mathbb{R}^d$. Then, for each cluster $j \in \{1, ..., k\}$, we compute the fraction of training examples with positive labels in that cluster:

$$y^{(j)} = \frac{\sum_{i=1}^n \mathbb{1}(k_i = j) \cdot y_i}{\sum_{i=1}^n \mathbb{1}(k_i = j)},$$

where $k_i = \arg\min_{j \in \{1,...,k\}} \|x_i - x^{(j)}\|_2^2$ is the cluster assigned to x_i.

(a) (1 pt) If $k = 1$, what does the resulting model family look like? (In other words, what functions are possible across all possible datasets?) Constant function

(b) (1 pt) If $k \to \infty$, what does the resulting model family look like? Arbitrary function

(c) (2 pt) Does increasing k increase, decrease, or not affect variance? Increases

4. (8 pts) Let ϕ_1 and ϕ_2 be two feature maps over inputs $x \in \mathbb{R}^d$, and consider the linear regression models $\beta_1^\top \phi_1(x)$ and $\beta_2^\top \phi_2(x)$ corresponding to ϕ_1 and ϕ_2, respectively. For each of the following, can the variance of $\beta_1^\top \phi_1(x)$ be higher than, lower than, or either higher than or lower than that of $\beta_2^\top \phi_2(x)$? Indicate all possibilities. Unless otherwise specified, assume no regularization.

(a) (1 pt) ϕ_1 has strictly more features than ϕ_2. [Hint: What if the features in ϕ_1 are all the same?] either

(b) (1 pt) The features in ϕ_1 are a strict superset of those in ϕ_2 (e.g., ϕ_2 consists of quadratic features, and ϕ_1 consists of quadratic features and some others). higher

(c) (1 pt) ϕ_1 and ϕ_2 contain exactly the same features, and we use L_2 regularization for $\beta_1^\top \phi_1(x)$ but not for $\beta_2^\top \phi_2(x)$. lower

(d) (1 pt) The features in ϕ_1 are a strict superset of those in ϕ_2, and we use L_2 regularization for $\beta_1^\top \phi_1(x)$ but not for $\beta_2^\top \phi_2(x)$. either

(e) (2 pts) We construct $\phi_1(x)$ by using principal components analysis on the training inputs $\{x_i\}_{i=1}^n$, and taking the projection onto the top k components. We construct ϕ_2 similarly, but take the top k' components, where $k' < k$. higher

(f) (2 pts) We take $\phi_1(x)$ to be a bag of words model (i.e., each feature is an indicator $(\phi_1(x))_i = \mathbb{1}(w_i \in x)$ of whether word w_i is in sentence x), and take $\phi_2(x)$ to be bigram model (i.e., each feature is an indicator $(\phi_2(x))_i = \mathbb{1}(w_i w_i' \in x)$ of whether words w_i and w_i' occur sequentially in sentence x). lower

5. (4 pts) Suppose we use AdaBoost to train an ensemble of logistic regression models over a feature map ϕ. For each of the following hyperparameters, indicate whether increasing it tends to increase or decrease variance (you should give exactly one answer for each part).

(a) (1 pt) The number of T iterations of AdaBoost (equivalently, the number of base models in the final ensemble) increases

(b) (1 pt) Assuming we use L_2 regularization, the magnitude of λ (recall that the regularization term is $\lambda \cdot \|\beta\|_2^2$, where β are the logistic regression parameters) decreases

(c) (1 pt) The number of training examples n (i.e., the training dataset is $\{(x_i, y_i)\}_{i=1}^n$) decreases

(d) (1 pt) The number of features d (i.e., each feature vector is $\phi(x) \in \mathbb{R}^d$) increases

6. (4 pts) For which of the following algorithms is optimization perfect—i.e., the standard optimizer is guaranteed to find the model that globally minimizes the loss function?

   (a) (1 pt) Logistic regression, if the loss is the NLL (a.k.a. cross-entropy loss) yes

   (b) (1 pt) Logistic regression, if the loss is the accuracy no

   (c) (1 pt) Neural network with one hidden layer, if the loss is the NLL no

   (d) (1 pt) k-means clustering, if the loss is the squared distance to the centroid representing each point, averaged over points no

7. (4 pts) Consider a logistic regression model, which has likelihood function

$$p_\theta(Y = y \mid X = x) = \begin{cases} \sigma(\theta^\top x) & \text{if } y = 1 \\ 1 - \sigma(\theta^\top x) & \text{if } y = 0, \end{cases}$$

where $\sigma(z) = \frac{1}{1+e^{-z}}$ is the sigmoid function. Suppose we have already fit the parameters θ, and we want to rescale the predicted probabilities. One strategy for doing so is *temperature scaling*, where we introduce an additional real-valued parameter $\beta \in \mathbb{R}$, and consider

$$p_\beta(Y = y \mid X = x) = \begin{cases} \sigma(\beta \cdot \theta^\top x) & \text{if } y = 1 \\ 1 - \sigma(\beta \cdot \theta^\top x) & \text{if } y = 0. \end{cases}$$

   (a) (2 pts) What happens to the classification boundary if we take $\beta \to 0$ (i.e., very small but not quite zero)? What happens to the predicted probabilities (i.e., what values can they take)? Classification boundary does not change, probabilities $\to 1/2$

   (b) (2 pts) What happens to the classification boundary if we take $\beta \to \infty$? What happens to the predicted probabilities? Classification boundary does not change, probabilities $\to 0$ or $\to 1$ (just $\to 1$ is fine)

8. (10 pts) Consider a neural network with one hidden layer:

$$f_W(x) = \begin{bmatrix} 1 \\ 1 \end{bmatrix}^\top \sigma\left(\begin{bmatrix} W_{11} & W_{12} \\ W_{21} & W_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \right),$$

where $\sigma(z)$ is some nonlinear function. Note that

$$f_W(x) = \sigma(W_{11}x_1 + W_{12}x_2) + \sigma(W_{21}x_1 + W_{22}x_2).$$

(a) (4 pts) What is the gradient $\nabla_W f_W(x)$? In particular, compute each partial derivative $\frac{\partial}{\partial W_{ij}} f_W(x)$; then, the gradient is

$$\nabla_W f_W(x) = \begin{bmatrix} \frac{\partial}{\partial W_{11}} f_W(x) & \frac{\partial}{\partial W_{12}} f_W(x) \\ \frac{\partial}{\partial W_{21}} f_W(x) & \frac{\partial}{\partial W_{22}} f_W(x) \end{bmatrix}.$$

You can leave your answer in terms of $\sigma(z)$ and $\sigma'(z) = \frac{\partial}{\partial z}\sigma(z)$. We have

$$\nabla_W f_W(x) = \begin{bmatrix} \sigma'(W_{11}x_1 + W_{12}x_2)x_1 & \sigma'(W_{11}x_1 + W_{12}x_2)x_2 \\ \sigma'(W_{21}x_1 + W_{22}x_2)x_1 & \sigma'(W_{21}x_1 + W_{22}x_2)x_2 \end{bmatrix}$$

(b) (2 pts) What is the gradient $\nabla_W L(W; x, y)$ of the loss $L(W; x, y) = (f_W(x) - y)^2$? You do not need to expand $f_W(x)$ (but you should expand $\nabla_W f_W(x)$). We have

$$\begin{aligned} \nabla_W L(W; x, y) &= 2(f_W(x) - y)\nabla_W f_W(x) \\ &= 2(f_W(x) - y) \begin{bmatrix} \sigma'(W_{11}x_1 + W_{12}x_2)x_1 & \sigma'(W_{11}x_1 + W_{12}x_2)x_2 \\ \sigma'(W_{21}x_1 + W_{22}x_2)x_1 & \sigma'(W_{21}x_1 + W_{22}x_2)x_2 \end{bmatrix} \end{aligned}$$

(c) (2 pts) Suppose the parameters satisfy $W_{11} = W_{21}$ and $W_{12} = W_{22}$. After one step gradient descent (with learning rate η), do these equalities still hold? In other words, recalling that the gradient descent update rule is

$$W' \leftarrow W - \eta \cdot \nabla_W L(W; x, y),$$

where $\eta \in \mathbb{R}_{>0}$ is the learning rate, show that $W'_{11} = W'_{21}$ and $W'_{12} = W'_{22}$. Note that the updated weights are

$$\begin{aligned} \begin{bmatrix} W'_{11} & W'_{12} \\ W'_{21} & W'_{22} \end{bmatrix} &= \begin{bmatrix} W_{11} & W_{12} \\ W_{21} & W_{22} \end{bmatrix} - \eta \cdot 2(f_W(x) - y) \begin{bmatrix} \sigma'(W_{11}x_1 + W_{12}x_2)x_1 & \sigma'(W_{11}x_1 + W_{12}x_2)x_2 \\ \sigma'(W_{21}x_1 + W_{22}x_2)x_1 & \sigma'(W_{21}x_1 + W_{22}x_2)x_2 \end{bmatrix} \\ &= \begin{bmatrix} W_{11} & W_{12} \\ W_{11} & W_{12} \end{bmatrix} - \eta \cdot 2(f_W(x) - y) \begin{bmatrix} \sigma'(W_{11}x_1 + W_{12}x_2)x_1 & \sigma'(W_{11}x_1 + W_{12}x_2)x_2 \\ \sigma'(W_{11}x_1 + W_{12}x_2)x_1 & \sigma'(W_{11}x_1 + W_{12}x_2)x_2 \end{bmatrix} \end{aligned}$$

Thus, we have $W'_{11} = W'_{21}$ and $W'_{12} = W'_{22}$.

(d) (2 pts) Based on your answer, briefly explain why initializing the weight matrix to zero (i.e., $W_{11} = W_{12} = W_{21} = W_{22} = 0$) is a bad idea. We always have $W_{11} = W_{21}$ and $W_{12} = W_{22}$, so the neural network effectively has half as many parameters.

9. (10 pts) Consider two binary random variables X_1, X_2.

(a) (3 pts) There are three possible Bayesian networks over these two random variables; draw all three of them. We have $X_1 \to X_2$, $X_2 \to X_1$, and $X_1 \quad X_2$

(b) (3 pt) For each possible Bayesian network, indicate whether it can represent joint distributions of the form $p(X_1 = x_1, X_2 = x_2) = p(X_1 = x_1)p(X_2 = x_2)$. All three of them can

(c) (3 pt) For each possible Bayesian network, indicate whether it can represent an arbitrary joint distribution $p(X_1 = x_1, X_2 = x_2)$. Only $X_1 \to X_2$ and $X_2 \to X_1$

(d) (1 pt) We say two Bayesian networks are *equivalent* if they can represent exactly the same class (a.k.a. subset) of possible joint distributions. Indicate which pairs of Bayesian networks you drew are equivalent. $X_1 \to X_2$ and $X_2 \to X_1$ are equivalent.

10. (10 pts) In class, we learned that recurrent neural networks (RNNs) can be viewed as reusing the same parameter across layers. In this problem, we will examine the gradients of RNNs via a toy example.

(a) (4 pts) Consider a neural network $y = f_\theta(x)$, where $x \in \mathbb{R}$, $y \in \mathbb{R}$, and $\theta \in \mathbb{R}^2$, where

$$f_\theta(x) = \theta_2 \sigma(\theta_1 x),$$

for some nonlinear function $\sigma(z)$. What is the gradient $\nabla_\theta f_\theta(x) = \begin{bmatrix} \frac{\partial}{\partial \theta_1} f_\theta(x) & \frac{\partial}{\partial \theta_2} f_\theta(x) \end{bmatrix}^\top$? You can leave your answer in terms of σ and σ', where $\sigma'(z) = \frac{\partial}{\partial z}(z)$. We have

$$\nabla_\theta f_\theta(x) = \begin{bmatrix} \theta_2 \sigma'(\theta_1 x) x \\ \sigma(\theta_1 x) \end{bmatrix}$$

(b) (4 pts) Consider a neural network $y = h_\beta(x)$, where $x \in \mathbb{R}$, $y \in \mathbb{R}$, and $\beta \in \mathbb{R}$, where

$$h_\beta(x) = \beta \sigma(\beta x),$$

with σ is as before. What is the gradient $\nabla_\beta h_\beta(x) = \frac{\partial}{\partial \beta} h_\beta(x)$? We have

$$\nabla_\beta h_\beta(x) = \beta \sigma'(\beta x) x + \sigma(\beta x)$$

(c) (2 pts) Note that letting $\theta = \begin{bmatrix} \beta & \beta \end{bmatrix}^\top$, then we have $h_\beta(x) = f_\theta(x)$. Using this fact, express the gradient $\nabla_\beta h_\beta(x)$ in terms of $\nabla_\theta f_\theta(x)$. [Hint: Use the chain rule to compute $\frac{\partial}{\partial \beta} f_{[\beta \;\; \beta]^\top}(x)$.] Check to make sure your answer is consistent with the previous parts! We have

$$\begin{aligned} \nabla_\beta h_\beta(x) = \frac{\partial}{\partial \beta} f_{[\beta \;\; \beta]^\top}(x) &= \frac{\partial}{\partial \beta} \begin{bmatrix} \beta \\ \beta \end{bmatrix}^\top \nabla_\theta f_\theta(x) \\ &= \begin{bmatrix} 1 \\ 1 \end{bmatrix}^\top \nabla_\theta f_\theta(x) \end{aligned}$$

11. (10 pts) Consider the following Markov decision process with states $S = \{s_1, s_2, ..., s_n\}$ and actions

$$A = \{a_1 = \text{move left}, a_2 = \text{move right}\}.$$

The transitions are deterministic: Suppose the agent is currently in state s_i. Then, taking action a_1 transitions the agent to state s_{i-1} (unless $i = 1$, in which case it stays in s_1), and taking a_2 transitions it to s_{i+1} (unless $i = n$, in which case it stays in s_n). Finally, the rewards are

$$R(s_i) = \begin{cases} 1 & \text{if } i = 1 \\ 0 & \text{if } i \in \{2, 3, ..., n-1\} \\ n + 10 & \text{if } i = n, \end{cases}$$

the discount factor is $\gamma = 1$, the time horizon is $T = n$, and the initial state is s_1. Suppose we are running a reinforcement learning algorithm, and it knows all the MDP transitions, as well as the rewards for all states except s_n.

(a) (2 pts) Write down the optimal policy—i.e., the action $\pi^*(s_i) \in A$ to take for each i. What is its cumulative expected reward? $\pi(s) = a_2$ for all s, cumulative reward of $n + 11$

(b) (2 pts) Suppose we act randomly in this MDP—i.e., choose action $a \sim \text{Uniform}(\{a_1, a_2\})$ i.i.d. on each step. What is the probability of reaching state s_n (from initial state s_1) in a single rollout within the time horizon? $(1/2)^{n-1}$ (give one point for $(1/2)^n$)

(c) (2 pts) Suppose that our current estimate the reward of s_n to be $R(s_n) = 0$. Write down the optimal policy $\hat{\pi}(s_i) \in A$ for each i for this estimate. $\pi(s_i) = a_1$

(d) (2 pts) Recall that an ϵ-greedy policy acts randomly with probability ϵ and optimally based on the current estimate (given in part (c)) with probability $1 - \epsilon$. What is the probability that an ϵ-greedy policy based on $\hat{\pi}$ reaches s_n (from initial state s_1) in a single rollout within the time horizon? $(\epsilon/2)^{n-1}$ (give one point for $(\epsilon/2)^n$)

(e) (2 pts) Based on your above answers, briefly explain why random exploration (including ϵ-greedy) will perform poorly for learning the unknown reward $R(s_n)$. The probability of exploring state s_n is exponentially small.