6.390: Midterm Exam, Fall 2024

Do not tear exam booklet apart!

- This is a closed book exam. One page (8 1/2 in. by 11 in.) of notes, front and back, are permitted. Calculators are not permitted.
- The total exam time is 2 hours.
- The problems are not necessarily in any order of difficulty.
- Record all your answers in the places provided. If you run out of room for an answer, continue on a blank page and mark it clearly.
- If a question seems vague or under-specified to you, make an assumption, write it down, and solve the problem given your assumption.
- If you have a question, raise your hand or come to the front of the room.
- **Write your name on every piece of paper.**

Name: ______________________ MIT Email: ______________________

Question	Points	Score
1	24	
2	12	
3	14	
4	12	
5	20	
6	18	
Total:	100	

Name: ____________________

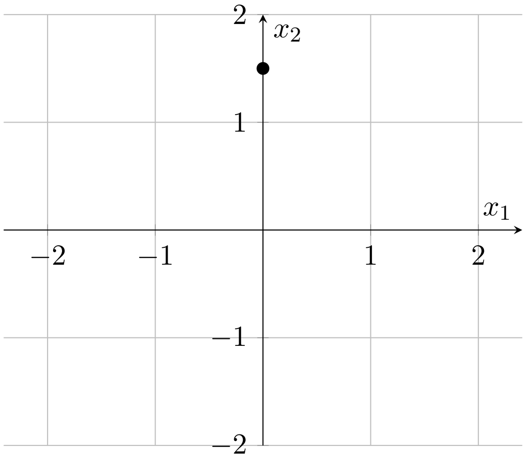
Classification

1. (24 points) Recall that a linear logistic classifier is characterized by

$$h(x; \theta, \theta_0) = \sigma(\theta^T x + \theta_0),$$

where $\sigma(\cdot)$ is the standard sigmoid function, $\sigma(z) = 1/(1 + \exp(-z))$. Define the argument of the sigmoid function in $h(\cdot)$ to be $z = \theta^T x + \theta_0$.

(a) i. On the graph below, draw the linear separator defined by the parameters $\theta = [-2, -2]^T$, $\theta_0 = 2$. Be sure to include a direction normal pointing in the direction of the positive class.



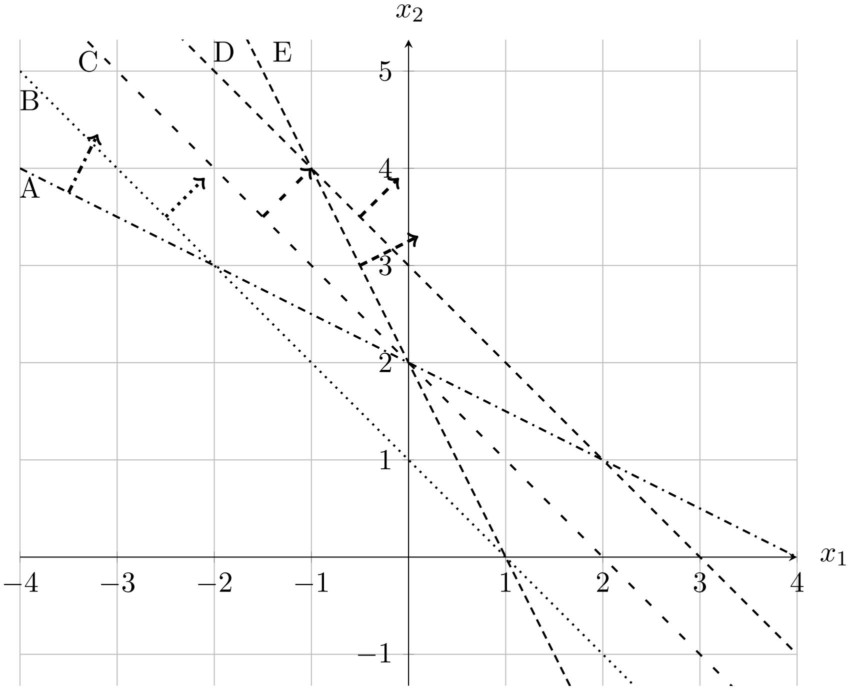
ii. Would the corresponding linear logistic classifier assign the indicated point at $(x_1, x_2) = (0.0, 1.5)$ as positive or negative?

Circle one: Positive Negative

iii. What value of z does the classifier assign to the indicated point? (Your answer should be a specific number.)

iv. What is the numerical probability output by the classifier? (You can leave a mathematical expression involving e, but your solution must not have matrix operations left to be done.)

(b) The following plot represents a two-dimensional space into which five separating hyperplanes for classifiers have been drawn, each with an associated normal vector intended to point toward the positive examples.



For each of the hypotheses parameterized by the values of the values of (θ, θ_0) below, identify the matching hyperplane from the plot above.

i. $\theta = \begin{bmatrix} 2 \\ 2 \end{bmatrix}$, $\theta_0 = -2$

Circle one: A B C D E

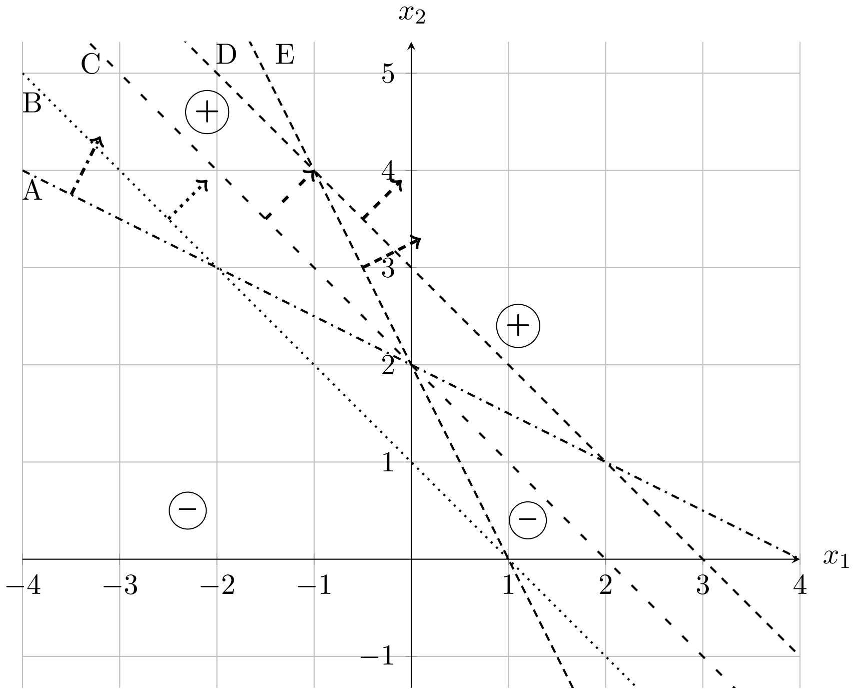
ii. $\theta = \begin{bmatrix} 2 \\ 2 \end{bmatrix}$, $\theta_0 = -6$

Circle one: A B C D E

iii. $\theta = \begin{bmatrix} 2 \\ 4 \end{bmatrix}$, $\theta_0 = -8$

Circle one: A B C D E

(c) The plot below is the same as from the previous part, but, now with some "held-out" data points that were not used in training the models. Each data point is labeled as positive (+) or negative (−).



Imagine the linear logistic classifier model with parameter $|\theta_1| = 1$ that corresponds to each of the drawn, oriented separating hyperplanes.

i. What is the accuracy of each of the models (A–E) on the set of held out data indicated?

Accuracy of $A =$ $B =$

$C =$ $D =$

$E =$

ii. Now using a NLL measure of Loss, which model has the minimum loss on the held out data? Justify your response. Hint: you do not need to explicitly compute the NLL value.

Circle one: A B C D E

Justification:

(d) Now, we would like to analyze a multiclass linear logistic classifier with $K = 3$ classes. For this part of the problem, we are still working with only 2 input features (x_1, x_2), but we choose to fold θ_0 into the θ matrix by adding a row to the bottom of the θ matrix representing the θ_0's and we add a 1 to the end of each x column vector. So, in this framing, let x be a data point, $x = [x_1, x_2, 1]^T$. Our θ will be a 3×3 matrix and let $z = \theta^T x$ be a 3×1 vector with $z = [z_1, z_2, z_3]^T$. Then, the output of the model will be defined as,

$$g = \text{softmax}(z) = \begin{bmatrix} \exp(z_1)/\sum_{i=1}^{3}\exp(z_i) \\ \exp(z_2)/\sum_{i=1}^{3}\exp(z_i) \\ \exp(z_3)/\sum_{i=1}^{3}\exp(z_i) \end{bmatrix}.$$

Recall that the vector g represents the likelihood assigned to each of the three classes, and the class prediction is the made from the largest element of g.

i. Suppose that we have a model defined by the following matrix:

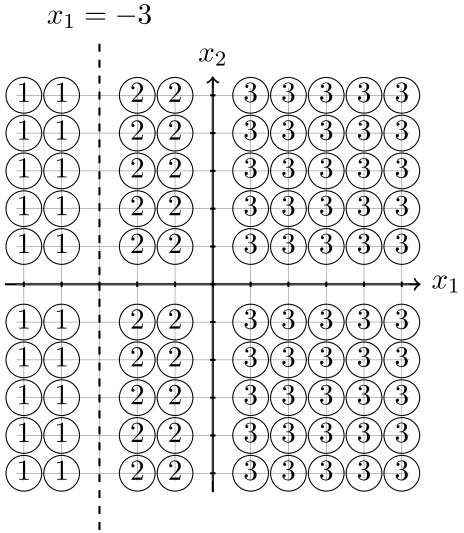
$$\theta = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}.$$

Consider the data point $x = [1, -1, 1]^T$. Compute $z = \theta^T x$ and determine which class will be assigned to x.

$z =$

Class assigned:

ii. Examine the classification problem represented by the graph below, where points are labelled with their class: $1, 2,$ or 3. Also given below is a model represented by the matrix θ (note that this is θ and so the first column represents $\theta_1, \theta_2,$ and θ_0 for class 1).



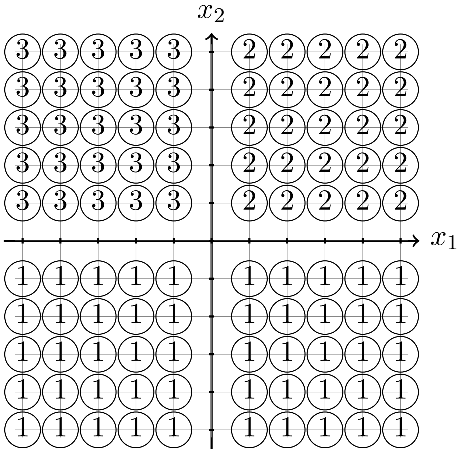
$$\theta = \begin{bmatrix} -1 & 0 & 1 \\ 0 & 0 & 0 \\ -3 & 0 & 0 \end{bmatrix}$$

Does the provided model defined by θ perfectly separate the data as desired in the graph above? If yes, show your reasoning. If no, identify all data points in the graph above that are misclassified.

Circle one: Yes No

Explanation:

iii. Examine the classification problem represented by the graph below, where points are labelled with their class: $1, 2$, or 3. Also given below is a model represented by the matrix θ (note that this is θ and so the first column represents $\theta_1, \theta_2,$ and θ_0 for class 1).



$$\theta = \begin{bmatrix} 0 & 1 & -1 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Does the provided model defined by θ perfectly separate the data as desired in the graph above? If yes, show your reasoning. If no, identify all data points in the graph above that are misclassified.

Circle one: Yes No

Explanation:

Feathery Featurizations

2. (12 points) In a popular bird-themed board game, players collect bird cards, each representing a different species. Each card has several attributes relevant to the game mechanics, such as the bird's wingspan, the type of nest it builds, the maximum number of eggs it can hold, and the food required to play the card. To use these bird cards in a machine learning model, these attributes need to be transformed into numerical features that the model can interpret.

   In this problem, you will determine how to encode the features of these birds and encode specific examples. Describe how you would encode each of the features below for use in a machine learning model and the dimensions of the encoded feature. Consider different types of encoding (e.g., binary, one-hot encoding, thermometer, normalization, etc.) and explain your reasoning for each feature.

   (a) Nest Type: Birds can build different types of nests (bowl, cavity, platform, or ground). A bird can only have one of these types of nests or a "wildcard" nest, meaning that their nest counts for any and all types of nests.

   Write the encoding of: (1) a bird that builds a bowl nest and (2) a bird that builds a wildcard nest.

   (b) Habitat: Each bird can live in one or more of three habitats: forest, grassland, or wetland.

   Write the encoding of: a bird that can live in the forest or grasslands.

   (c) Wingspan: A continuous feature representing the bird's wingspan (in centimeters). The minimum wingspan is 0 cm and maximum is roughly 300 cm.

   Write the encoding of: a bird with a wingspan of 50cm.

(d) Egg Limit: Each bird card specifies how many eggs the bird can hold, represented as an integer. The minimum number of eggs is zero and maximum number of eggs is eight.

Write the encoding of: a bird that can hold 4 eggs.

(e) Food Cost: Each bird requires a specific combination of food to play the card. There are 5 types of food (e.g., Invertebrate, Seed, Fruit, Fish, Rodents) and birds cost no more than three food. Example to encode: A bird that costs 2 Fruit and 1 Seed to play.

(f) Points: Each bird card is worth a certain number of points, which is an integer value (a minimum of 0 and maximum of 10).

Write the encoding of: a bird worth 7 points.

Random Descent

3. (14 points) Your friend Jordan is interested in learning algorithms for producing linear regression models. However, they have resolved to not compute any gradients. Instead, they decide to come up with their own iterative learning algorithm, Random Descent. At every iteration, we randomly decide to increase or decrease the parameter values by the learning rate. If this change results in a lower mean-squared error (MSE), the update is accepted; otherwise, the parameters remain unchanged. Pseudo-code for learning two parameters, a, b, with Random Descent is as follows:

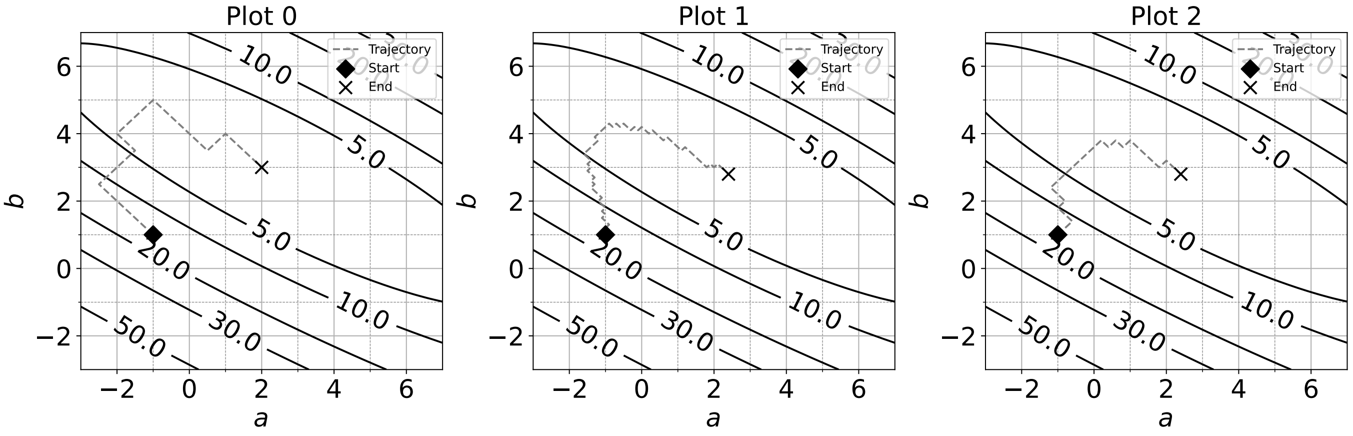
```
def random_descent(a, b, X, Y, max_iter, learning_rate, decay=1):
    # Compute initial error
    error = MSE(a, b, X, Y)

    # Iterative loop for random descent
    for iter in range(max_iter):
        # Propose new values for a and b
        a_new = a + coin_flip([-1, 1]) * learning_rate
        b_new = b + coin_flip([-1, 1]) * learning_rate

        # Compute new error
        new_error = MSE(a_new, b_new, X, Y)

        # Accept new parameters if error decreases
        if new_error < error:
            a = a_new
            b = b_new
            error = new_error
            learning_rate = learning_rate*decay

    # Return final learned parameters
    return a, b
```

Here, `coin_flip` is used to randomly pick whether to increase or decrease the parameter value with equal probability. Suppose that Jordan has some data set $\mathcal{D} = \{(x^{(i)}, y^{(i)})\}_{i=1}^{100}$ where each $x^{(i)} \in \mathbb{R}^2$ is a 2-dimensional feature vector and $y^{(i)} \in \mathbb{R}$ is its corresponding label. Each column of `X` is a feature vector, and each "column" of `Y` is the corresponding target output value. The `error` is then computed to be

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^{n} \left(y^{(i)} - h(x^{(i)}; \{a, b\}) \right)^2 .$$

(a) In Jordan's first data set, the feature vectors take the form $x^{(i)} = [x_1^{(i)}, 1]^T$. Their hypotheses take the form $h_1(x; \{a, b\}) = ax_1 + b$. Jordan runs three different instances of Random Descent, each initialized with `a=-1, b=1, max_iter=1000, decay=1`, and using learning rates 0.1, 0.2, and 0.5. However, they lost track of which rate corresponds to which of the three instances.

In each panel below is a contour plot showing lines of constant `error`, visualizing the trajectory of one instance of minimizing the MSE with random descent:



Match each of the three plots with the learning rate that was used to generate it. Each choice of learning rate was used exactly once.

Plot 0: ______________

Plot 1: ______________

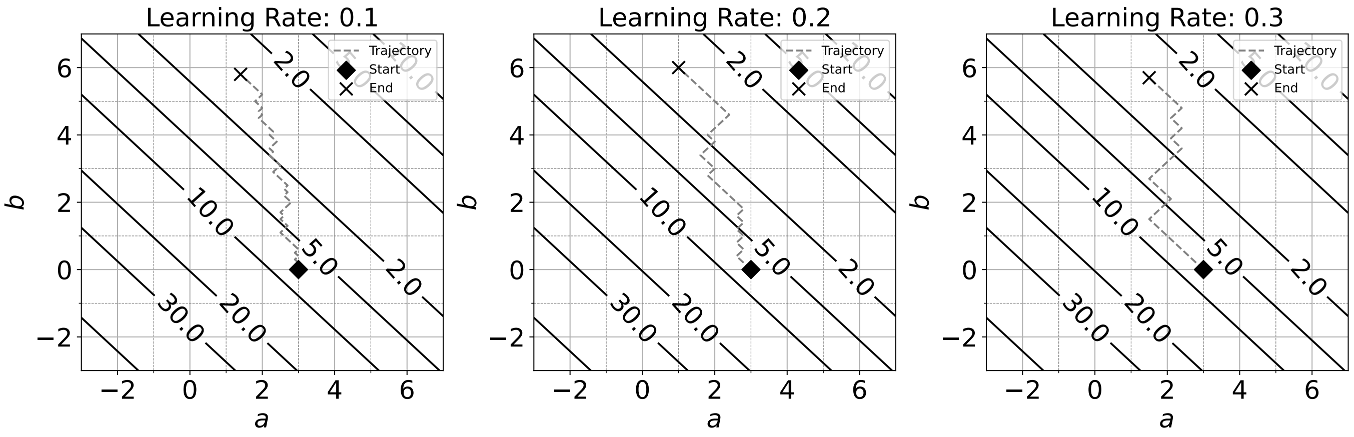
Plot 2: ______________

(b) Consider the left-most plot in Part (a). What are the learned parameter values for a, b? Round a, b to the closest integer.

a: ______________

b: ______________

(c) In Jordan's second data set the feature vectors take the form $x^{(i)} = [x_1^{(i)}, x_2^{(i)}]^T$. Their hypotheses take the form $h_2(x; \{a, b\}) = ax_1 + bx_2$. Jordan runs three instances of Random Descent, each initialized with `a=3, b=0, max_iter=1000, decay=1`, but with different learning rates. Three contour plots showing lines of constant `error` are shown below, corresponding to these three runs, with the learning rate as indicated in the title:



An oracle approaches, who claims to be omniscient. She tells Jordan that the best hypothesis takes the form $h(x) = 4x_1 + 3x_2$.

Does this claim have any merit? Should we trust this oracle? Mark all that are true and explain your reasoning.

○ The results of our experiments are enough to disprove the oracle's claim. Random Descent learned the uniquely best model.

○ Based on the contour plots, parameter values of $a = 4, b = 3$ would also correspond to a hypothesis which minimizes the MSE.

○ Given more iterations, every instance of Random Descent ran on this data set would eventually converge to $a = 4, b = 3$.

○ We need access to a larger training data set so that we can be more confident in our learned parameter values.

○ We need to evaluate the hypotheses on a validation data set.

Reasoning:

(d) Jordan would like to start using a `decay` with the learning rate such that progressively smaller steps will be taken at each iteration. They would like to utilize 4-fold cross validation in order to find a `decay` value that produces hypotheses that generalize to held-out validation data.

Fill in the blanks in the pseudocode below to implement 4-fold cross validation.

```
decays = [0.8,0.9,0.99,0.999]

for i = 1 to 4:

    Divide X,Y into ___________________________________________________

    for j = 1 to 4:

        a[j],b[j] = random_descent(0,0,_______,______,1000,0.3,__________)

        error[j] = _______________________________

    avg_error[i] = _______________________________

best_decay = _______________________________
```

Name: ____________________________

Logistic Mysteries

4. (12 points) The standard stochastic gradient descent algorithm is defined as

$$\Theta^{(t)} = \Theta^{(t-1)} - \eta(t)\nabla_\Theta f_{i(t)}(\Theta^{(t-1)}) \ (t = 1, 2, 3, \dots).$$

Here, $\Theta^{(t)} = [\theta^{(t)}, \theta_0^{(t)}]^T \in \mathbb{R}^2$ are two-dimensional vectors. Our objective is the negative log-likelihood function,

$$f_k(\Theta) = \mathcal{L}_{nll}(h(x^{(k)}, \Theta), y^{(k)}), \quad h\left(x, \begin{bmatrix}\theta \\ \theta_0\end{bmatrix}\right) = \sigma(\theta x + \theta_0).$$

$\sigma(\cdot)$ is the standard sigmoid function. At each iteration t, $i(t)$, is an integer selected randomly from $\{1, 2, \dots, n\}$. A variable learning rate $\eta(t) > 0$ is used.

Stochastic gradient descent was applied to minimize (with respect to Θ) the (non-regularized) logistic regression objective function

$$J(\Theta) = \sum_{k=1}^{n} f_k(\Theta)$$

for the training dataset $\{(x^{(j)}, y^{(j)})\}_{j=1}^n$ containing n training samples $(x^{(j)}, y^{(j)})$ where $x^{(j)} \in \mathbb{R}$ are input samples, and $y^{(j)} \in \{0, 1\}$ are the corresponding labels, for $j \in \{1, 2, \dots, n\}$. The resulting first few values of Θ are:

$$\Theta^{(0)} = \begin{bmatrix}3 \\ 0\end{bmatrix}, \quad \Theta^{(1)} = \begin{bmatrix}0 \\ 2\end{bmatrix}, \quad \Theta^{(2)} = \begin{bmatrix}2 \\ 3\end{bmatrix}, \quad \Theta^{(3)} = \begin{bmatrix}2 \\ 2\end{bmatrix}.$$

Recall that

$$\frac{\partial}{\partial\theta} f_k(\Theta) = x^{(k)}(h(x^{(k)}, \Theta) - y^{(k)}), \quad \frac{\partial}{\partial\theta_0} f_k(\Theta) = h(x^{(k)}, \Theta) - y^{(k)}.$$

(a) Find $x^{(i(1))}, x^{(i(2))}, x^{(i(3))}$. Show your reasoning.

$x^{i(1)} =$ $\qquad$ $x^{i(2)} =$ $\qquad$ $x^{i(3)} =$

For your convenience, we repeat that the resulting first few values of Θ are:

$$\Theta^{(0)} = \begin{bmatrix} 3 \\ 0 \end{bmatrix}, \quad \Theta^{(1)} = \begin{bmatrix} 0 \\ 2 \end{bmatrix}, \quad \Theta^{(2)} = \begin{bmatrix} 2 \\ 3 \end{bmatrix}, \quad \Theta^{(3)} = \begin{bmatrix} 2 \\ 2 \end{bmatrix}.$$

(b) Find $y^{(i(1))}, y^{(i(2))}, y^{(i(3))}$. Show your reasoning. Hint: recall that $y^{(j)} \in \{0, 1\}$.

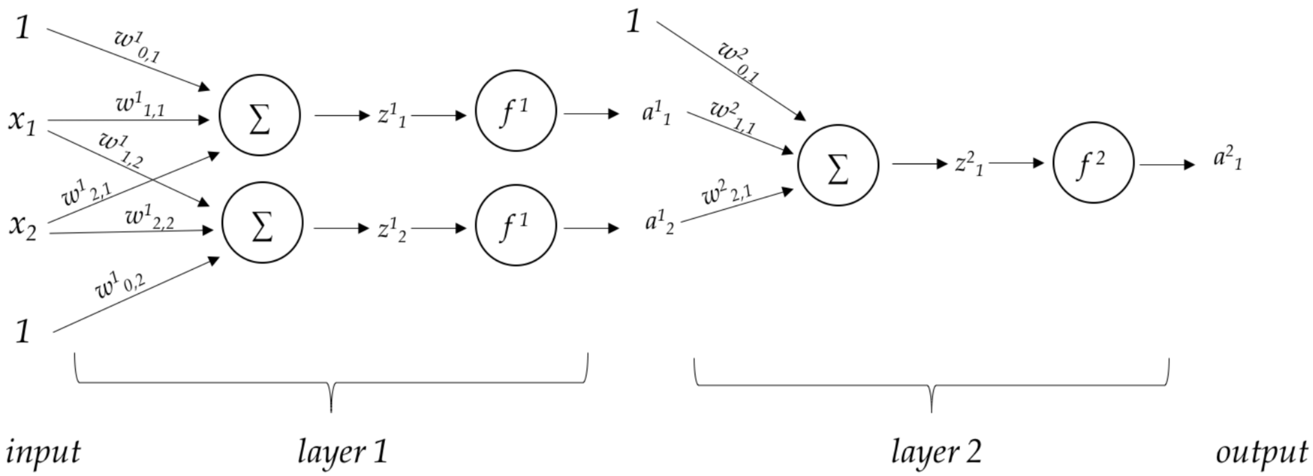
$y^{i(1)} =$ $\qquad$ $y^{i(2)} =$ $\qquad$ $y^{i(3)} =$

Slippery Slope

5. (20 points) Alice, Bob, and Carl are debating the utility of non-linear activation functions:

- Alice believes that ReLUs rule! (After all, it's right there in the name.)
- Bob thinks that we should always use sigmoids.
- Carl wants to invent their own activation function.

Consider the following neural network:

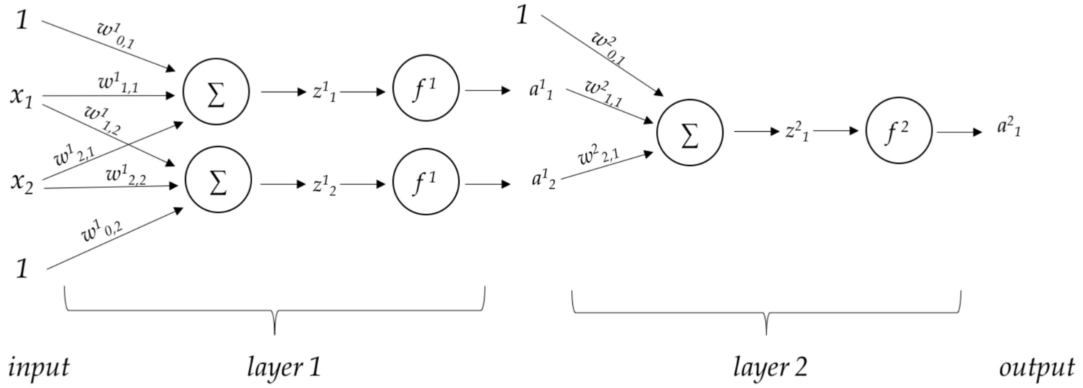


(a) We follow Alice's suggestion and keep layer-1 activation as ReLUs. Recall that the ReLU function is defined as $f(z) := \max(z, 0)$. Assume that we let the derivative of ReLU at $z = 0$ be zero. For this part, answer with a real number; if you believe the set up isn't enough to determine a real-valued answer, answer "it depends".

i. If $a_1^1 = 0.5$, what is $\frac{\partial a_1^1}{\partial z_1^1}$?

ii. If $a_1^1 = 0.5$, what is $\frac{\partial a_1^1}{\partial x_1}$?

iii. If $a_1^1 = 0$, what is $\frac{\partial a_1^1}{\partial x_1}$?



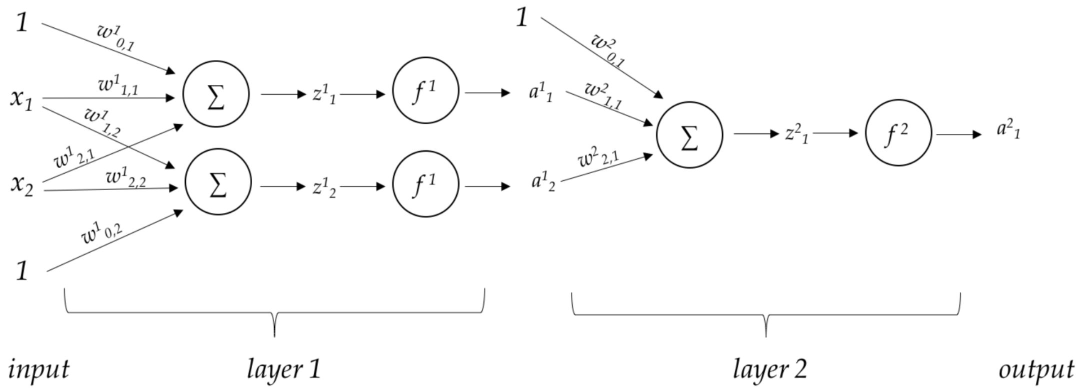
(b) We now follow Alice's preference to have the layer-2 activation as ReLUs as well. Suppose the current weights are $w^2_{0,1} = 1, w^2_{1,1} = 1, w^2_{2,1} = 1$. The final output a^2_1 is 0. Using a step-size of 0.1, what would be the updated values of these weights?

(c) Suppose instead we follow Bob's desire to keep layer-1 activation as sigmoids. Recall that the sigmoid function is defined as $f(z) := \frac{1}{1+\exp(-z)}$. For this part, answer with a real number; if you believe the setup isn't enough to determine a real-valued answer, answer "it depends".

i. If $a^1_1 = 0.5$, what is $\frac{\partial a^1_1}{\partial z^1_1}$?

ii. If $a^1_1 = 0.2$, what is $\frac{\partial a^1_1}{\partial x_1}$?

iii. If $a^1_1 = 0.8$, what is $\frac{\partial a^1_1}{\partial x_1}$?



(d) Carl argues that both sigmoid and ReLU suffer from having a large portion of the input space where the gradients are zero (or almost zero) – which can make backpropagation very difficult.

i. Suppose that we used sigmoids in our network (reproduced above for your convenience). In particular, during an SGD update, these sigmoid units may have (nearly) zero gradient, that is,

$$\frac{\partial a^i_j}{\partial z^i_j} \approx 0$$

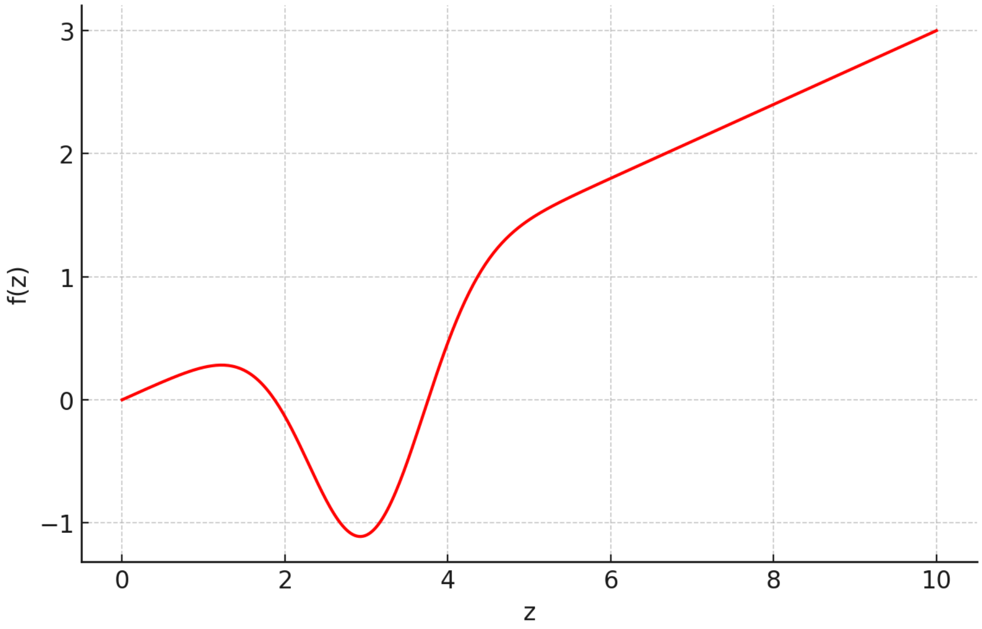
where $i, j = 1, 2$. Would you agree with Carl's argument that having near-zero gradients could be troublesome for learning? Explain your reasoning.

ii. Alice and Bob argue that in reality people use either sigmoids or ReLU for good reason, and that the scenario that Carl described do not happen that frequently in practice. What might be the reason? Explain your reasoning.

(e) Carl proposes a "Nike"-like swoosh type of activation function, defined as

$$f(z) = -2 \cdot e^{-1 \cdot (z-3)^2} + 0.3 \cdot z$$

and graphed below:



If we use this activation function in the first layer, and let $z^1 = [z_1^1, z_2^1]^T$ be the first layer pre-activation output, and $a^1 = [a_1^1, a_2^1]^T$ be the first layer post-activation output, what would be $\frac{\partial a^1}{\partial z^1}$? (We are looking for a symbolic answer only, not a specific number.)

Judging by the First Steps

6. (18 points) Let $f : \mathbb{R} \to \mathbb{R}$ be an *unknown* function. Consider the following additional assumptions (A)-(C) one *could* make about $f(\cdot)$:

(A) $f(\cdot)$ is differentiable everywhere;

(B) $f(\cdot)$ is differentiable everywhere and convex;

(C) $f(\cdot)$ is the regularized linear regression objective function

$$f(\theta) = \lambda\theta^2 + \frac{1}{n}\sum_{k=1}^{n}(\theta x_k - y_k)^2$$

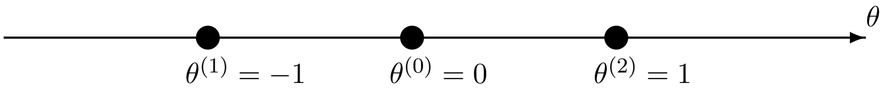
for some $\lambda > 0$, $n \in \{1, 2, 3, \dots\}$, and real $x_1, \dots, x_n$, $y_1, \dots, y_n$.

The standard gradient descent algorithm

$$\theta^{(t)} = \theta^{(t-1)} - \eta\nabla_\theta f(\theta^{(t-1)}) \quad (t = 1, 2, \dots)$$

with initial guess $\theta^{(0)} = 0$ and some positive, fixed learning rate $\eta > 0$ is applied to try to find an argument of minimum of $f(\cdot)$ numerically. Knowing that the first two steps of this algorithm have resulted in $\theta^{(1)} = -1$ and $\theta^{(2)} = 1$, as shown below, what can be learned about $f(\cdot)$ based on this information?

t	$\theta^{(t-1)}$	$\theta^{(t)}$
1	0	-1
2	-1	1
3	1	?



(a) First, consider assumption (C). As the objective function is quadratic, its gradient $\nabla_\theta f(\theta)$ will be a linear function of the form $\nabla_\theta f(\theta) = \theta a + b$. Write down a and b in terms of the data $\{x_i, y_i\}_{i=1}^n$ and regularization hyperparameter λ.

$a =$

$b =$

(b) Find the set Θ_3 of all possible values of $\theta^{(3)}$.

i. under assumption (C):
Hint: Your answer should be a real number.

$\Theta_3 =$

ii. under assumption (B):
Hint: Use the fact that the derivative of a convex scalar function is non-decreasing.

$\Theta_3 =$

iii. under assumption (A):

$\Theta_3 =$

(c) Find the set $\Theta_{\min}$ of all possible values of the argument of minimum of f (if it has one).

i. under assumption (C):
Hint: Your answer should be a real number.

$\Theta_{\min} =$

ii. under assumption (B):
Hint: You may want to try drawing a rough sketch of $\nabla_\theta f(\theta)$ from the information you have.

$\Theta_{\min} =$

iii. under assumption (A):

$\Theta_{\min} =$

(d) Is it possible that, as the sequence of gradient descent steps continues starting from $\theta^{(0)} = 0$, $\theta^{(1)} = -1$, $\theta^{(2)} = 1$, that $\theta^{(t)}$ will converge to the argument of minimum of $f(\cdot)$ as $t \to +\infty$?

i. under assumption (C):

○ Possible

○ Impossible

ii. under assumption (B):

○ Possible

○ Impossible

iii. under assumption (A):

○ Possible

○ Impossible

Name: ______________________

Work space

Name: ______________________________

Work space