

10-601 Machine Learning  
Spring 2024  
Exam 2 Practice Problems  
Updated: March 20, 2024  
Time Limit: N/A

---

Name:  
AndrewID:

**Instructions:**

- Fill in your name and Andrew ID above. Be sure to write neatly, or you may not receive credit for your exam.
  - Clearly mark your answers in the allocated space **on the front of each page**. If needed, use the back of a page for scratch space, but you will not get credit for anything written on the back of a page. If you have made a mistake, cross out the invalid parts of your solution, and circle the ones which should be graded.
  - No electronic devices may be used during the exam.
  - Please write all answers in pen.
  - You have N/A to complete the exam. Good luck!
-

## Instructions for Specific Problem Types

For “Select One” questions, please fill in the appropriate bubble completely:

**Select One:** Who taught this course?

- Henry Chai
- Marie Curie
- Noam Chomsky

If you need to change your answer, you may cross out the previous answer and bubble in the new answer:

**Select One:** Who taught this course?

- Henry Chai
- Marie Curie
- Noam Chomsky

For “Select all that apply” questions, please fill in all appropriate squares completely:

**Select all that apply:** Which are scientists?

- Stephen Hawking
- Albert Einstein
- Isaac Newton
- I don't know

Again, if you need to change your answer, you may cross out the previous answer(s) and bubble in the new answer(s):

**Select all that apply:** Which are scientists?

- Stephen Hawking
- Albert Einstein
- Isaac Newton
- I don't know

For questions where you must fill in a blank, please make sure your final answer is fully included in the given space. You may cross out answers or parts of answers, but the final answer must still be within the given space.

**Fill in the blank:** What is the course number?

10-601

10-~~7~~601

# 1 Optimization

1. **Select all that apply:** Which of the following are correct regarding Gradient Descent (GD) and stochastic gradient descent (SGD)?

- Each update step in SGD pushes the parameter vector closer to the parameter vector that minimizes the objective function.
- The gradient computed in SGD is, in expectation, equal to the gradient computed in GD.
- The gradient computed in GD has a higher variance than that computed in SGD, which is why in practice SGD converges faster in time than GD.
- None of the above.

2. (a) **Select all that apply:** Determine if the following 1-D functions are convex. Assume that the domain of each function is  $\mathbb{R}$ . The definition of a convex function is as follows:

$$f(x) \text{ is convex} \iff f(\alpha x + (1 - \alpha)z) \leq \alpha f(x) + (1 - \alpha)f(z), \forall \alpha \in [0, 1] \text{ and } \forall x, z.$$

- $f(x) = x + b$  for any  $b \in \mathbb{R}$
- $f(x) = c^2x$  for any  $c \in \mathbb{R}$
- $f(x) = ax^2 + b$  for any  $a \in \mathbb{R}$  and any  $b \in \mathbb{R}$
- $f(x) = 0$
- None of the above

(b) **Select all that apply:** Consider the convex function  $f(z) = z^2$ . Let  $\alpha$  be our learning rate in gradient descent.

For which values of  $\alpha$  will  $\lim_{t \rightarrow \infty} f(z^{(t)}) = 0$ , assuming the initial value of  $z$  is  $z^{(0)} = 1$  and  $z^{(t)}$  is the value of  $z$  after the  $t$ -th iteration of gradient descent?

- $\alpha = 0$
- $\alpha = \frac{1}{2}$
- $\alpha = 1$
- $\alpha = 2$
- None of the above

(c) **Numerical answer:** Give the range of all values for  $\alpha \geq 0$  such that  $\lim_{t \rightarrow \infty} f(z^{(t)}) = 0$ , assuming the initial value of  $z$  is  $z^{(0)} = 1$ .

## 2 Logistic Regression and Regularization

1. A generalization of logistic regression to a multiclass settings involves expressing the per-class probabilities  $P(y = c|x)$  as the softmax function  $\frac{\exp(w_c^T x)}{\sum_{d \in C} \exp(w_d^T x)}$ , where  $c$  is some class from the set of all classes  $C$ .

Consider a 2-class problem (labels 0 or 1). Rewrite the above expression for this situation to end up with expressions for  $P(Y = 1|x)$  and  $P(Y = 0|x)$  that we have already come across in class for binary logistic regression.

2. Given a training set  $\mathcal{D} = \{(\mathbf{x}^{(1)}, y^{(1)}), \dots, (\mathbf{x}^{(N)}, y^{(N)})\}$  where  $\mathbf{x}^{(i)} \in \mathbb{R}^d$  is a feature vector and  $y_i \in \{0, 1\}$  is a binary label, we want to find the parameters  $\hat{w}$  that maximize the likelihood for the training set, assuming a parametric model of the form

$$p(y = 1|x; w) = \frac{1}{1 + \exp(-w^T x)}.$$

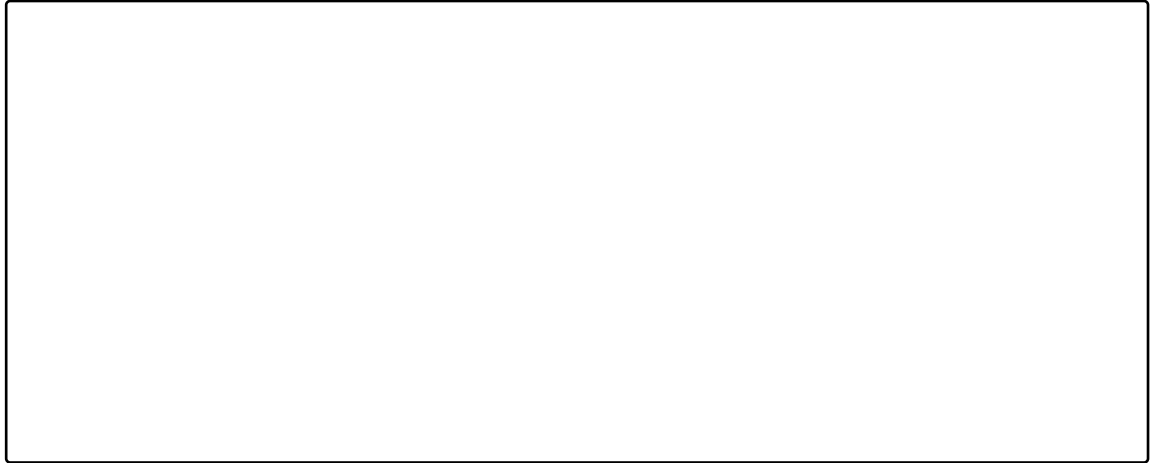
The conditional log likelihood of the training set is

$$\ell(w) = \sum_{i=1}^n y_i \log p(y_i, |x_i; w) + (1 - y_i) \log(1 - p(y_i, |x_i; w)),$$

and the gradient is

$$\nabla \ell(w) = \sum_{i=1}^n (y_i - p(y_i|x_i; w))x_i.$$

- a) Is it possible to get a closed form for the parameters  $\hat{w}$  that maximize the conditional log likelihood? How would you compute  $\hat{w}$  in practice?



- b) For a binary logistic regression model, we predict  $y = 1$  when  $p(y = 1|x) \geq 0.5$ . Show that this is a linear classifier.



- c) Consider the case with binary features, i.e,  $x \in \{0, 1\}^d$ , where feature  $x_1$  is rare and happens to appear in the training set with only label 1. What is  $\hat{w}_1$ ? Is the gradient ever zero for any finite  $w$ ? Why is it important to include a regularization term to control the norm of  $\hat{w}$ ?

3. Given the following dataset,  $\mathcal{D}$ , and a fixed parameter vector,  $\boldsymbol{\theta}$ , write an expression for the binary logistic regression conditional likelihood.

$$\mathcal{D} = \{(\mathbf{x}^{(1)}, y^{(1)} = 0), (\mathbf{x}^{(2)}, y^{(2)} = 0), (\mathbf{x}^{(3)}, y^{(3)} = 1), (\mathbf{x}^{(4)}, y^{(4)} = 1)\}$$

- Write your answer in terms of  $\boldsymbol{\theta}$ ,  $\mathbf{x}^{(1)}$ ,  $\mathbf{x}^{(2)}$ ,  $\mathbf{x}^{(3)}$ , and  $\mathbf{x}^{(4)}$ .
- Do not include  $y^{(1)}$ ,  $y^{(2)}$ ,  $y^{(3)}$ , or  $y^{(4)}$  in your answer.
- Don't try to simplify your expression.

**Conditional likelihood:**

4. Write an expression for the decision boundary of binary logistic regression with a bias term for two-dimensional input features  $x_1 \in \mathbb{R}$  and  $x_2 \in \mathbb{R}$  and parameters  $b$  (the intercept parameter),  $w_1$ , and  $w_2$ . Assume that the decision boundary occurs when  $P(Y = 1 \mid \mathbf{x}, b, w_1, w_2) = P(Y = 0 \mid \mathbf{x}, b, w_1, w_2)$ .

- (a) Write your answer in terms of  $x_1$ ,  $x_2$ ,  $b$ ,  $w_1$ , and  $w_2$ .

**Decision boundary equation:**

- (b) What is the geometric shape defined by this equation?

5. We have now feature engineered the two-dimensional input,  $x_1 \in \mathbb{R}$  and  $x_2 \in \mathbb{R}$ , mapping

it to a new input vector:  $\mathbf{x} = \begin{bmatrix} 1 \\ x_1^2 \\ x_2^2 \end{bmatrix}$

- (a) Write an expression for the decision boundary of binary logistic regression with this feature vector  $\mathbf{x}$  and the corresponding parameter vector  $\boldsymbol{\theta} = [b, w_1, w_2]^T$ . Assume that the decision boundary occurs when  $P(Y = 1 | x, \boldsymbol{\theta}) = P(Y = 0 | x, \boldsymbol{\theta})$ . Write your answer in terms of  $x_1$ ,  $x_2$ ,  $b$ ,  $w_1$ , and  $w_2$ .

**Decision boundary expression:**

- (b) Assume that  $w_1 > 0$ ,  $w_2 > 0$ , and  $b < 0$ . What is the geometric shape defined by this equation?

- (c) If we add an L2 regularization term when learning  $[w_1, w_2]^T$ , what happens to the **parameters** as we increase the  $\lambda$  that scales this regularization term?

- (d) If we add an L2 regularization term when learning  $[w_1, w_2]^T$ , what happens to the **decision boundary shape** as we increase the  $\lambda$  that scales this regularization term?

6. **Short Answer:** Your friend is training a logistic regression model with ridge regularization, where  $\lambda$  is the regularization constant. They run cross-validation for  $\lambda = [0.01, 0.1, 1, 10]$  and compare train, validation and test errors. They choose  $\lambda = 0.01$  because that had the lowest *test* error.

However, you observe that the test error linearly increases from  $\lambda = 0.01$  to 10 and thus, there exists a value of  $\lambda < 0.01$  that gives a lower test error. You tell your friend that they should run the cross-validation for  $\lambda = [0.0001, 0.001, 0.01]$  to get the optimal model.

Do you think you did the right thing by giving your friend this suggestion? Briefly justify

your answer in 1-2 concise sentences.

---

---

---

---

---



### 3 Feature Engineering and Regularization

1. **Model Complexity:** In this question we will consider the effect of increasing the model complexity, while keeping the size of the training set fixed. To be concrete, consider a classification task on the real line  $\mathbb{R}$  with distribution  $D$  and target function  $c^* : \mathbb{R} \rightarrow \{\pm 1\}$ , and suppose we have a random sample  $S$  of size  $n$  drawn iid from  $D$ . For each degree  $d$ , let  $\phi_d$  be the feature map given by  $\phi_d(x) = (1, x, x^2, \dots, x^d)$  that maps points on the real line to  $(d + 1)$ -dimensional space.

Now consider the learning algorithm that first applies the feature map  $\phi_d$  to all the training examples and then runs logistic regression. A new example is classified by first applying the feature map  $\phi_d$  and then using the learned classifier.

- a) For a given dataset  $S$ , is it possible for the training error to increase when we increase the degree  $d$  of the feature map? **Please explain your answer in 1 to 2 sentences.**
- b) Briefly **explain in 1 to 2 sentences** why the true error first drops and then increases as we increase the degree  $d$ .

## 4 Neural Networks

1. Match the corresponding neural network component to its role in the neural network.

Cross-Entropy

ELU

Linear

Mean Absolute Error

Mean Squared Error

ReLU

Sigmoid

Softmax

Stochastic Gradient Descent

Tanh

Activation Function

Loss Function

Optimizer

Layer

2. Consider the neural network architecture shown above for a binary classification problem. The values for weights and biases are shown in the figure. We define:

$$a_1 = w_{11}x_1 + b_{11}$$

$$a_2 = w_{12}x_1 + b_{12}$$

$$a_3 = w_{21}z_1 + w_{22}z_2 + b_{21}$$

$$z_1 = \text{ReLU}(a_1)$$

$$z_2 = \text{ReLU}(a_2)$$

$$z_3 = \sigma(a_3), \sigma(x) = \frac{1}{1+e^{-x}}$$

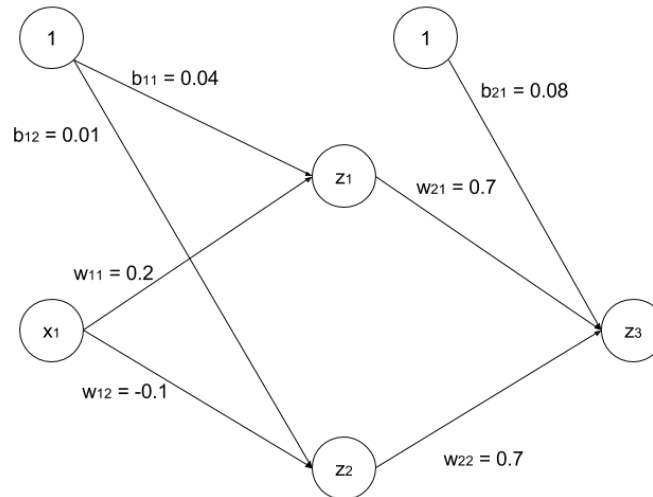


Figure 1: neural network

- (i) For  $x_1 = 0.3$ , compute  $z_3$  in terms of  $e$ .

- (ii) Which class does the network predict for the data point ( $x_1 = 0.3$ )? Note that  $\hat{y} = 1$  if  $z_3 > \frac{1}{2}$ , else  $\hat{y} = 0$ .

- (iii) Perform backpropagation on the bias term  $b_{21}$  by deriving the expression for the gradient of the loss function  $L(y, z_3)$  with respect to the bias term  $b_{21}$ ,  $\frac{\partial L}{\partial b_{21}}$ , in terms of the partial derivatives  $\frac{\partial \alpha}{\partial \beta}$ , where  $\alpha$  and  $\beta$  can be any of  $L, z_i, a_i, b_{ij}, w_{ij}, x_1$  for all valid values of  $i, j$ . Your backpropagation algorithm should be as explicit as possible — that is, make sure each partial derivative  $\frac{\partial \alpha}{\partial \beta}$  cannot be decomposed further into simpler partial derivatives. Do *not* evaluate the partial derivatives.

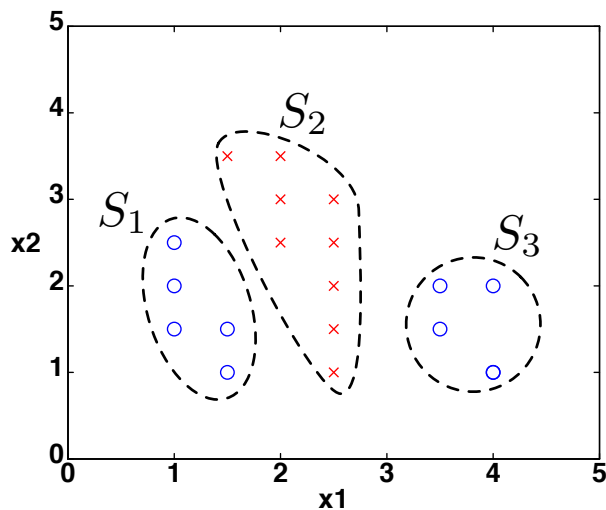
- (iv) Perform backpropagation on the bias term  $b_{12}$  by deriving the expression for the gradient of the loss function  $L(y, z_3)$  with respect to the bias term  $b_{12}$ ,  $\frac{\partial L}{\partial b_{12}}$ , in terms of the partial derivatives  $\frac{\partial \alpha}{\partial \beta}$ , where  $\alpha$  and  $\beta$  can be any of  $L, z_i, a_i, b_{ij}, w_{ij}, x_1$

for all valid values of  $i, j$ . Your backpropagation algorithm should be as explicit as possible — that is, make sure each partial derivative  $\frac{\partial \alpha}{\partial \beta}$  cannot be decomposed further into simpler partial derivatives. Do *not* evaluate the partial derivatives.

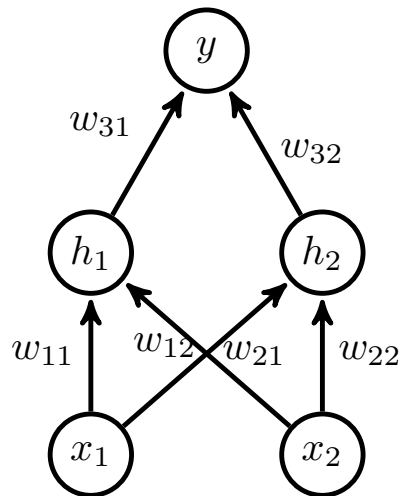


3. In this problem we will use a neural network to distinguish the crosses ( $\times$ ) from the circles ( $\circ$ ) in the simple data set shown in Figure 2a. Even though the crosses and circles are not linearly separable, we can break the examples into three groups,  $S_1$ ,  $S_2$ , and  $S_3$  (shown in Figure 2a) so that  $S_1$  is linearly separable from  $S_2$  and  $S_2$  is linearly separable from  $S_3$ . We will exploit this fact to design weights for the neural network shown in Figure 2b in order to correctly classify this training set. For all nodes, we will use the threshold activation function

$$\phi(z) = \begin{cases} 1 & z > 0 \\ 0 & z \leq 0. \end{cases}$$



(a) The data set with groups  $S_1$ ,  $S_2$ , and  $S_3$ .



(b) The neural network architecture

Figure 2

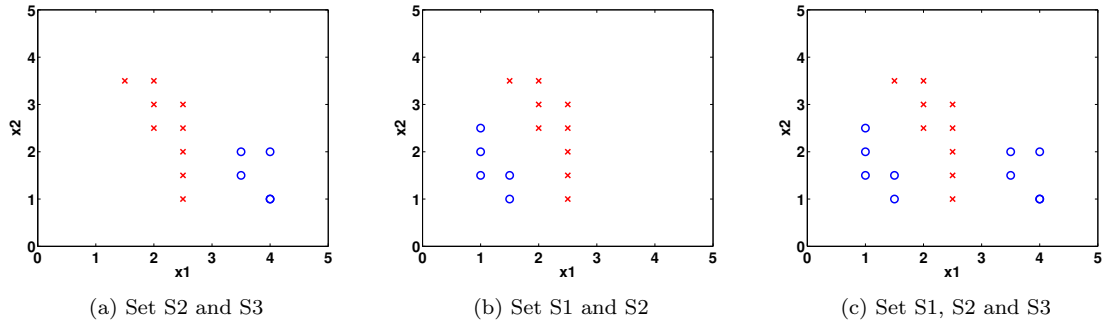


Figure 3: NN classification.

- (i) First we will set the parameters  $w_{11}$ ,  $w_{12}$  and  $b_1$  of the neuron labeled  $h_1$  so that its output  $h_1(x) = \phi(w_{11}x_1 + w_{12}x_2 + b_1)$  forms a linear separator between the sets  $S_2$  and  $S_3$ .

- (a) On Fig 3a, draw a linear decision boundary that separates  $S_2$  and  $S_3$ .
- (b) Write down the corresponding weights  $w_{11}$ ,  $w_{12}$ , and  $b_1$  so that  $h_1(x) = 0$  for all points in  $S_3$  and  $h_1(x) = 1$  for all points in  $S_2$ . One solution suffices and the same applies to (ii) and (iii).

- (ii) Next we set the parameters  $w_{21}$ ,  $w_{22}$  and  $b_2$  of the neuron labeled  $h_2$  so that its output  $h_2(x) = \phi(w_{21}x_1 + w_{22}x_2 + b_2)$  forms a linear separator between the sets  $S_1$  and  $S_2$ .

- (a) On Fig 3b, draw a linear decision boundary that separates  $S_1$  and  $S_2$ .
- (b) Write down the corresponding weights  $w_{21}$ ,  $w_{22}$ , and  $b_2$  so that  $h_2(x) = 0$  for all points in  $S_1$  and  $h_2(x) = 1$  for all points in  $S_2$ .

(iii) Now we have two classifiers  $h_1$  (to classify  $S_2$  from  $S_3$ ) and  $h_2$  (to classify  $S_1$  from  $S_2$ ). We will set the weights of the final neuron of the neural network based on the results from  $h_1$  and  $h_2$  to classify the crosses from the circles. Let  $h_3(x) = \phi(w_{31}h_1(x) + w_{32}h_2(x) + b_3)$ .

(a) Compute  $w_{31}, w_{32}, b_3$  such that  $h_3(x)$  correctly classifies the entire data set.

(b) Draw your decision boundary in Fig 3c.

4. One part of learning parameters in a neural network is getting the gradients of the parameters.

Suppose we have a dataset  $\mathcal{D}$  with  $N$  data points  $x_i$  with label  $y_i$ , where  $i \in [1, N]$ .  $x_i$  is a  $d \times 1$  vector and  $y_i \in \{0, 1\}$ . We use the data to train a neural network with one hidden layer:

$$h(x) = \sigma(W_1x + b_1)$$

$$p(x) = \sigma(W_2h(x) + b_2),$$

where  $\sigma(x) = \frac{1}{1 + \exp(-x)}$  is the sigmoid function,  $W_1$  is a  $n$  by  $d$  matrix,  $b_1$  is a  $n$  by 1 vector,  $W_2$  is a 1 by  $n$  matrix, and  $b_2$  is a 1 by 1 vector.

We use cross entropy loss and minimize the negative log likelihood to train the neural network:

$$\ell_{\mathcal{D}}(W) = \frac{1}{N} \sum_{i=1}^N \ell_i(W) = \frac{1}{m} \sum_{i=1}^N -(y_i \log p_i + (1 - y_i) \log(1 - p_i)),$$

where  $p_i = p(x_i), h_i = h(x_i)$ .

(a) Describe how you would derive the gradients w.r.t the parameters  $W_1, W_2$  and  $b_1, b_2$ . You do not need to write out the actual mathematical expression.

---



---

(b) When  $N$  is large, we typically use a small subset of the dataset to estimate the gradient — stochastic gradient descent (SGD). Explain why we use SGD instead of gradient descent.

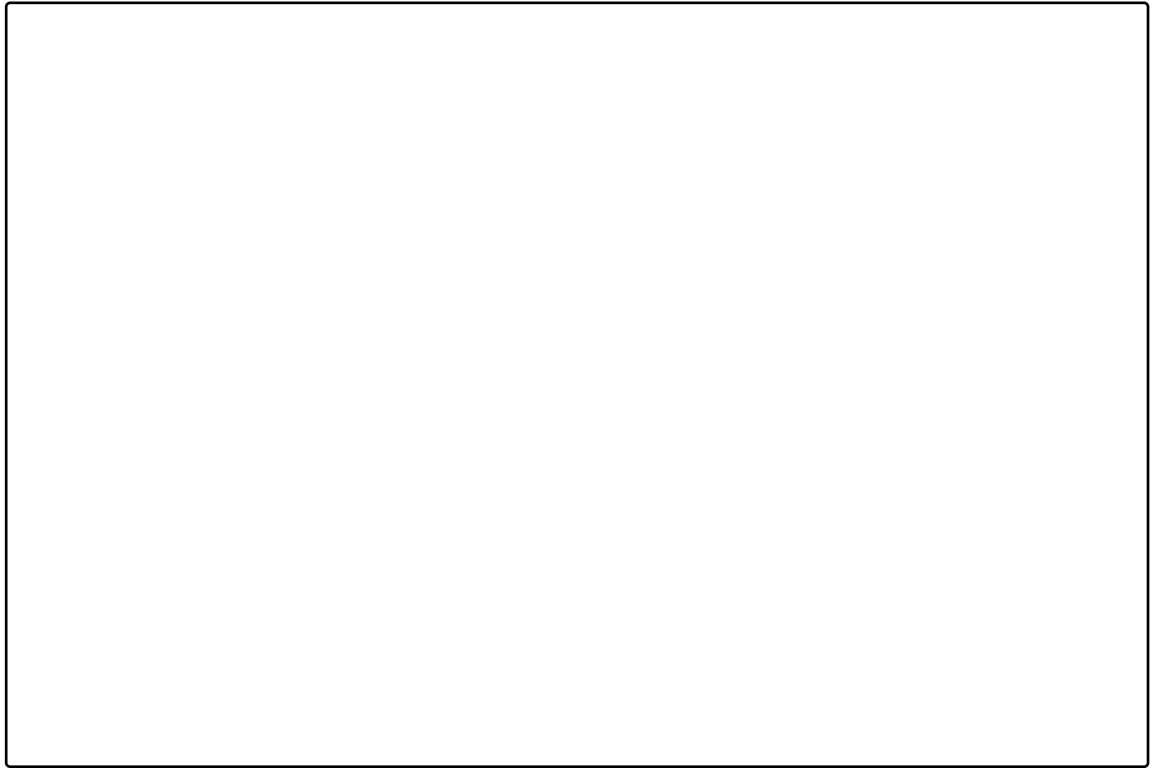
---



---

(c) Derive expressions for the following gradients:  $\frac{\partial \ell}{\partial p_i}, \frac{\partial \ell}{\partial W_2}, \frac{\partial \ell}{\partial b_2}, \frac{\partial \ell}{\partial h_i}, \frac{\partial \ell}{\partial W_1}, \frac{\partial \ell}{\partial b_1}$ . When deriving the gradient w.r.t. the parameters in lower layers, you may assume the

gradient in upper layers are available to you (i.e., you can use them in your equation). For example, when calculating  $\frac{\partial \ell}{\partial W_1}$ , you can assume  $\frac{\partial \ell}{\partial p_i}, \frac{\partial \ell}{\partial W_2}, \frac{\partial \ell}{\partial b_2}, \frac{\partial \ell}{\partial h_i}$  are known.



5. Consider the following neural network for a 2-D input,  $x_1 \in \mathbb{R}$  and  $x_2 \in \mathbb{R}$  where:

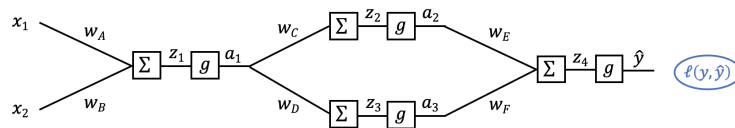


Figure 4: Neural Network

- All  $g$  functions are the same arbitrary non-linear activation function with no parameters
- $\ell(y, \hat{y})$  is an arbitrary loss function with no parameters, and:

$$z_1 = w_A x_1 + w_B x_2 \quad a_1 = g(z_1)$$

$$z_2 = w_C a_1 \quad a_2 = g(z_2)$$

$$z_3 = w_D a_1 \quad a_3 = g(z_3)$$

$$z_4 = w_E a_2 + w_F a_3 \quad \hat{y} = g(z_4)$$

**Note:** There are no bias terms in this network.

- (a) What is the chain of partial derivatives needed to calculate the derivative  $\frac{\partial \ell}{\partial w_E}$ ?

Your answer should be in the form:  $\frac{\partial \ell}{\partial w_E} = \overline{\frac{\partial?}{\partial?}} \overline{\frac{\partial?}{\partial?}} \dots$ . Make sure each partial derivative  $\frac{\partial?}{\partial?}$  in your answer cannot be decomposed further into simpler partial derivatives. **Do not evaluate the derivatives.** Be sure to specify the correct subscripts in your answer.

$$\frac{\partial \ell}{\partial w_E} =$$

- (b) The network diagram from above is repeated here for convenience: What is the

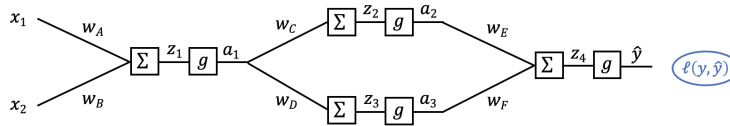


Figure 5: Neural Network

chain of partial derivatives needed to calculate the derivative  $\frac{\partial \ell}{\partial w_C}$ ?  
Your answer should be in the form:

$$\frac{\partial \ell}{\partial w_C} = \overline{\frac{\partial?}{\partial?}} \overline{\frac{\partial?}{\partial?}} \dots$$

Make sure each partial derivative  $\frac{\partial?}{\partial?}$  in your answer cannot be decomposed further into simpler partial derivatives. **Do not evaluate the derivatives.** Be sure to specify the correct superscripts in your answer.

$$\frac{\partial \ell}{\partial w_C} =$$

- (c) We want to modify our neural network objective function to add an L2 regularization term on the weights. The new objective is:

$$\ell(y, \hat{y}) + \lambda \frac{1}{2} \|w\|_2^2$$

where  $\lambda$  (lambda) is the regularization hyperparameter and  $\mathbf{w}$  is all of the weights in the neural network stacked into a single vector,  $\mathbf{w} = [w_A, w_B, w_C, w_D, w_E, w_F]^T$ .

Write the right-hand side of the new gradient descent update step for weight  $w_C$  given this new objective function. You may use  $\frac{\partial \ell}{\partial w_C}$  in your answer.

**Update:**  $w_C \leftarrow \dots$



6. Backpropagation in neural networks can lead to slow or unstable learning because of the vanishing or exploding gradients problem. Understandably, Neural the Narwhal does not believe this. To convince Neural, Lamar Jackson uses the example of an  $N$  layer neural network that takes in a scalar input  $x$ , and where each layer consists of a single neuron. More formally,  $x = o_0$ , and for each layer  $i \in \{1, 2, \dots, N\}$ , we have

$$s_i = w_i o_{i-1} + b_i$$

$$o_i = \sigma(s_i)$$

where  $\sigma$  is the sigmoid activation function. Note that  $w_i, b_i, o_i, s_i$  are all scalars.

- i. (1 point) Give an expression for  $\frac{\partial o_N}{\partial w_1}$ . Your expression should be in terms of the  $s_i$ 's, the  $w_i$ 's,  $N$ ,  $x_i$ , and  $\sigma'(\cdot)$ , the derivative of the sigmoid function.

- ii. (1 point) Knowing that  $\sigma'(\cdot)$  is at most  $\frac{1}{4}$  and supposing that all the weights are 1 (i.e.  $w_i = 1$  for all  $i$ ), give an upper bound for  $\frac{\partial o_N}{\partial w_1}$ . Your answer should be in terms of  $x$  and  $N$ .

7. Define a function  $\mathbf{floor} : \mathbb{R}_n \rightarrow \mathbb{R}_n$  such that

$$\mathbf{floor}(\mathbf{z}) = [z_i \text{ for } 0 \leq i \leq D]^T$$

or essentially, a function that produces an output vector by applying  $\lfloor \cdot \rfloor$  element-wise to the input vector.

Neural wants to use this function as an activation function to train his neural network. Is this possible? Explain why or why not.

---

---

---

## 5 Learning Theory

1. **True and Sample Errors:** Consider a classification problem with distribution  $D$  and target function  $c^* : \mathcal{R}^d \mapsto \pm 1$ . For any sample  $S$  drawn from  $D$ , answer whether the following statements are true or false, along with a brief explanation.

a) **True or False:** For a given hypothesis space  $\mathcal{H}$ , it is always possible to define a sufficient number of examples in  $S$  such that the true error is within a margin of  $\epsilon$  of the sample error for all hypotheses  $h \in H$  with a given probability.

---



---

b) **True or False:** The true error of any hypothesis  $h$  is an upper bound on its training error on the sample  $S$ .

---



---

2. Let  $X$  be the feature space and  $D$  be a distribution over  $X$ . We have a training data set

$$\mathcal{D} = \{(x_1, c^*(x_1)), \dots, (x_N, c^*(x_N))\},$$

$x_i$  i.i.d from  $D$ . We assume labels  $c^*(x_i) \in \{-1, 1\}$ .

Let  $\mathcal{H}$  be a hypothesis class and let  $h \in \mathcal{H}$  be a hypothesis. In this question we restrict ourselves to  $\mathcal{H}$ . We use

$$err_S(h) = \frac{1}{N} \sum_{i=1}^N \mathbb{I}(h(x_i) \neq c^*(x_i))$$

to denote the training error and

$$err_D(h) = P_{x \sim D}(h(x) \neq c^*(x))$$

to denote the true error. Recall that if the concept class is finite, in the realizable case

$$m \geq \frac{1}{\epsilon} \left[ \ln(|\mathcal{H}|) + \ln\left(\frac{1}{\delta}\right) \right]$$

labeled examples are sufficient so that with probability at least  $1 - \delta$ , all  $h \in \mathcal{H}$  with  $err_D(h) \geq \epsilon$  have  $err_S(h) > 0$ ; in the agnostic case,

$$m \geq \frac{1}{2\epsilon^2} \left[ \ln(|\mathcal{H}|) + \ln\left(\frac{2}{\delta}\right) \right]$$

labeled examples are sufficient such that with probability at least  $1 - \delta$ , all  $h \in \mathcal{H}$  have  $|err_D(h) - err_S(h)| < \epsilon$ .

- a) Briefly describe the difference between the realizable case and agnostic case.

---



---



---

- b) What is the full name of PAC learning? How do  $\epsilon$  and  $\delta$  tie into the name?

---



---



---

- c) **True or False:** Consider two finite hypothesis sets  $\mathcal{H}_1$  and  $\mathcal{H}_2$  such that  $\mathcal{H}_1 \subset \mathcal{H}_2$ . Let  $h_1 = \arg \min_{h \in \mathcal{H}_1} \text{err}_S(h)$  and  $h_2 = \arg \min_{h \in \mathcal{H}_2} \text{err}_S(h)$ . Because  $|\mathcal{H}_2| \geq |\mathcal{H}_1|$ ,  $\text{err}_D(h_2) \geq \text{err}_D(h_1)$ .

---



---



---

3. **Fill in the Blanks:** Complete the following sentence by circling one option in each square (options are separated by “/”s):

In order to prove that the VC-dimension of a hypothesis set  $\mathcal{H}$  is  $D$ , you must

show that  $\mathcal{H}$   shatter

of  $D$  data points and  shatter

of  $D + 1$  data points.

4. Consider the hypothesis set  $\mathcal{H}$  consisting of all positive intervals in  $\mathbb{R}$ , i.e. all hypotheses

$$\text{of the form } h(x; a, b) = \begin{cases} +1 & \text{if } x \in [a, b] \\ -1 & \text{if } x \notin [a, b] \end{cases}$$

- a) **Short Answer:** In 1-2 sentences, briefly justify why the VC dimension of  $\mathcal{H}$  is less than 3.

---



---



---

b) **Select one:** What is the VC dimension of  $\mathcal{H}$ ?

- 0
- 1
- 2

c) **Numerical Answer:** Now, consider hypothesis sets  $\mathcal{H}_k$  indexed by  $k$ , such that  $\mathcal{H}_k$  consists of all hypotheses formed by  $k$  **non-overlapping** positive intervals in  $\mathbb{R}$ . Give an expression for the VC dimension of  $\mathcal{H}_k$  in terms of  $k$ .

*Hint:* Think about how to repeatedly apply the result you found in Part (b).

5. **Select one:** Your friend, who is taking an introductory ML course, is preparing to train a model for binary classification. Having just learned about PAC Learning, she informs you that the model is in the finite, agnostic case.

Now she wants to know how changing certain values will change the number of labelled training data points required to satisfy the PAC criterion. For each of the following changes, determine whether the sample complexity will increase, decrease, or stay the same.

- i. (1 point) Using a simpler model (decreasing  $|\mathcal{H}|$ )
  - Sample complexity will increase
  - Sample complexity will decrease
  - Sample complexity will stay the same
- ii. (1 point) Choosing a new hypothesis set  $\mathcal{H}^*$ , such that  $|\mathcal{H}^*| = |\mathcal{H}|$ 
  - Sample complexity will increase
  - Sample complexity will decrease
  - Sample complexity will stay the same
- iii. (1 point) Decreasing  $\delta$ 
  - Sample complexity will increase
  - Sample complexity will decrease
  - Sample complexity will stay the same
- iv. (1 point) Decreasing  $\epsilon$ 
  - Sample complexity will increase
  - Sample complexity will decrease
  - Sample complexity will stay the same